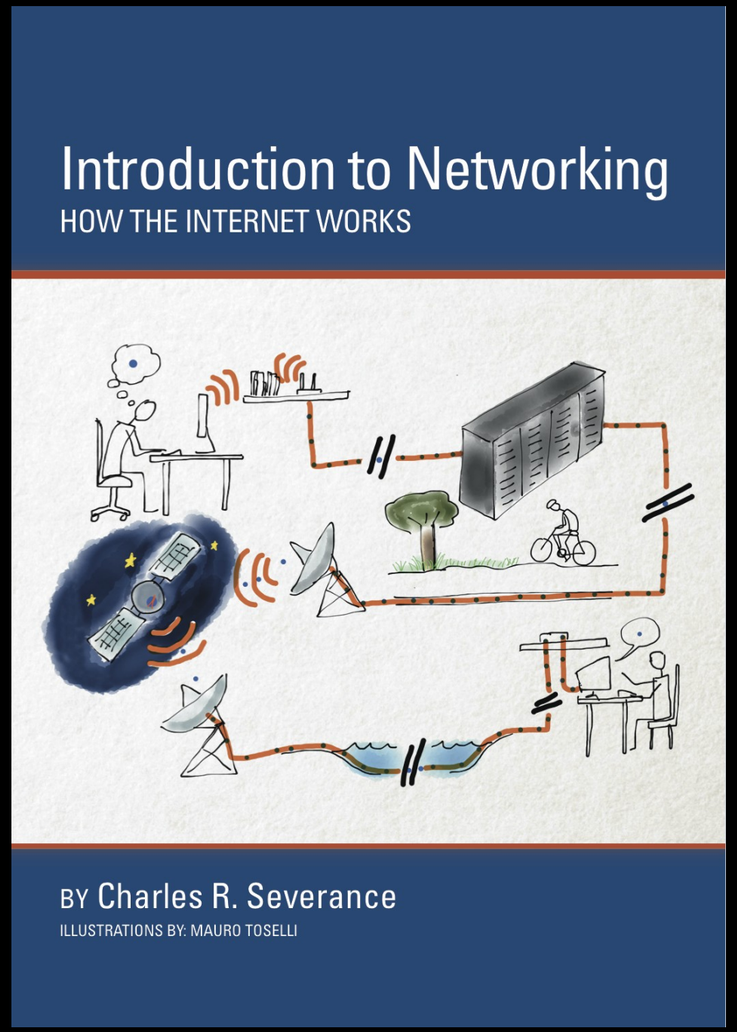
Сетевые программы

Глава 12

Пайтон для всех
www.py4e.com

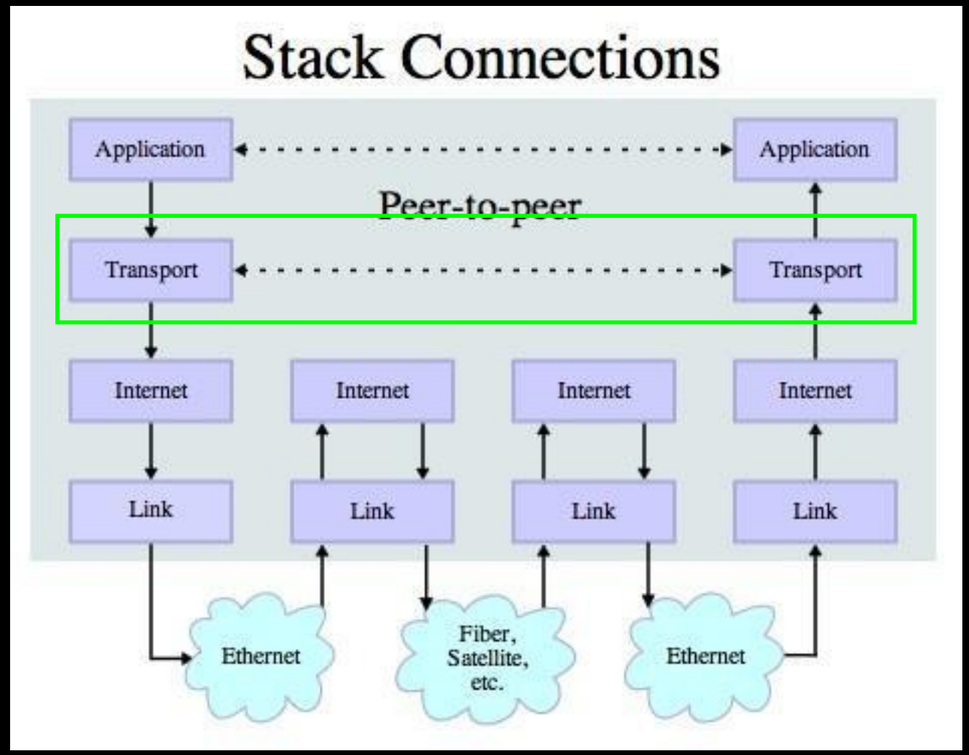
Бесплатная книга по сетевой архитектуре

- Если вам интересна данная тема и вы хотите знать больше
- www.net-intro.com



Протокол управления передачей (TCP)

- Создан на основе IP (Интернет протокол)
- Предполагается, что IP может потерять часть данных, поэтому TCP хранит и осуществляет повторный запрос данных в случае утери
- Осуществляет управление потоком с помощью «окна» передачи
- Обеспечивает наличие надежного туннеля



Source:
http://en.wikipedia.org/wiki/Internet_Protocol_Suite
https://ru.wikipedia.org/wiki/TCP/IP

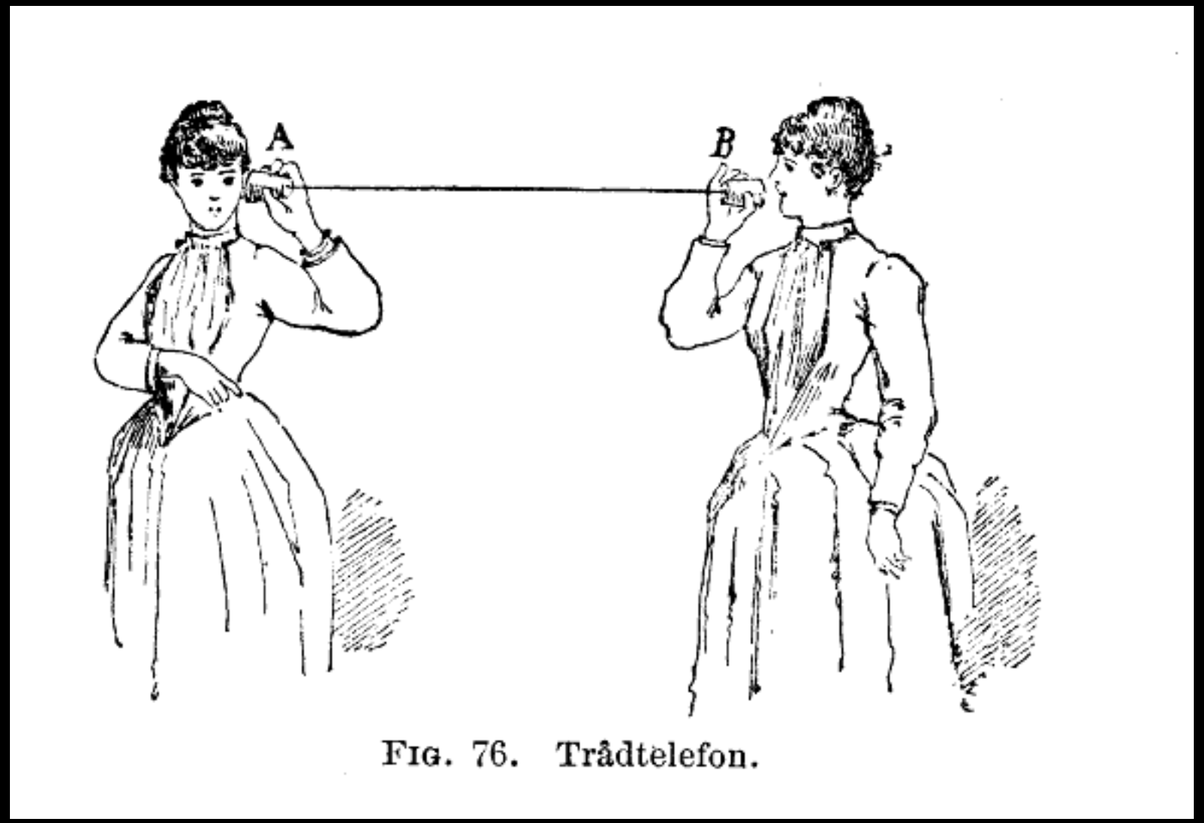


Fig. 76. Trådtelefon.

http://en.wikipedia.org/wiki/Tin_can_telephone

http://www.flickr.com/photos/kitcowan/2103850699/

TCP соединения / Сокеты

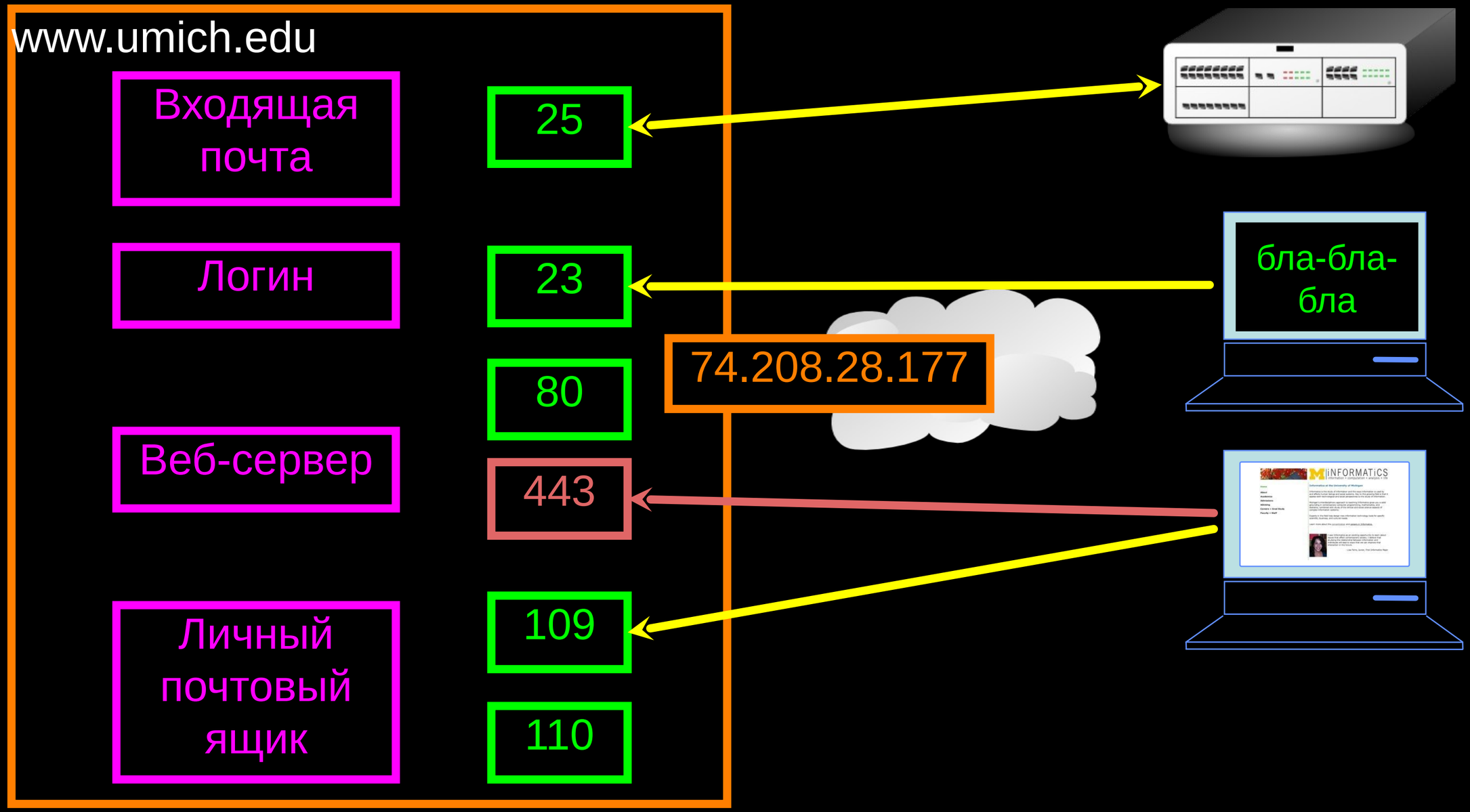
«В сетевой архитектуре Интернет-сокет или сетевой-сокет — это конечная точка двунаправленного меж-процессного потока обмена данными в компьютерной сети, основанной на Интернет Протоколе, такой как Интернет»

https://ru.wikipedia.org/wiki/Сокет_(программный_интерфейс)

Протокол TCP: Номера портов

- Порт — это конечная точка связи приложения или процесса
- Порт позволяет нескольким приложениям, использующим сеть, одновременно работать на одном сервере
- Существует список общепринятых номеров портов TCP

https://ru.wikipedia.org/wiki/Порт_(компьютерные_сети)

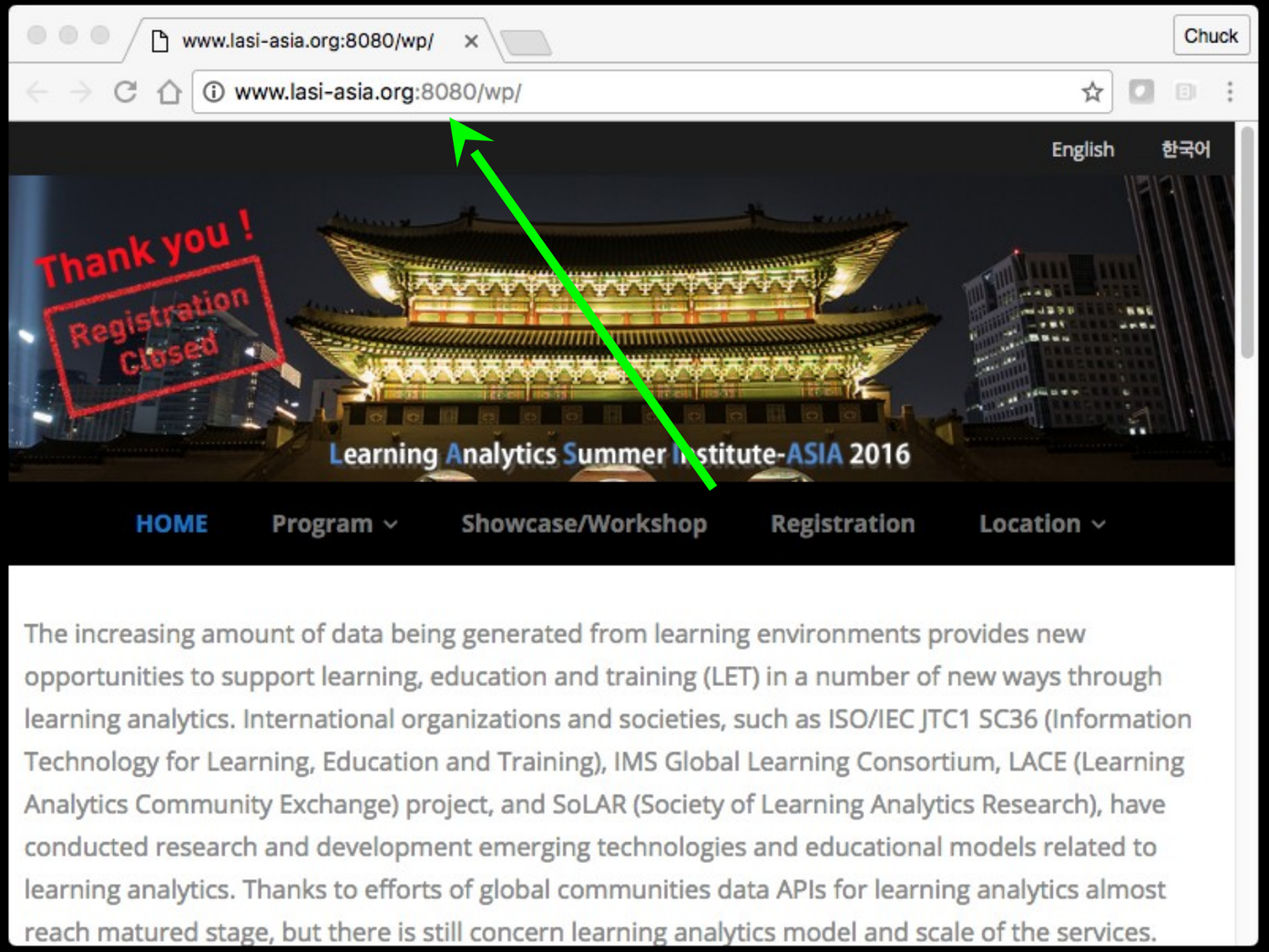


Клипарт: http://www.clker.com/search/networksym/1

Общеизвестные TCP-порты

- Telnet (23) - Login
- SSH (22) - Secure Login
- HTTP (80)
- HTTPS (443) - Secure
- SMTP (25) (Mail)
- IMAP (143/220/993) - Mail Retrieval
- POP (109/110) - Mail Retrieval
- DNS (53) - Domain Name
- FTP (21) - File Transfer

https://ru.wikipedia.org/wiki/Список_портов_TCP_и_UDP

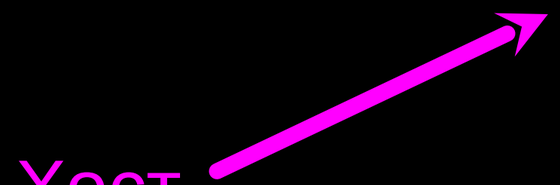


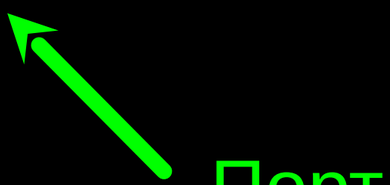
Иногда можно встретить номер порта в адресе, если сервер использует «нестандартный» порт

Сокеты в Пайтон

Пайтон имеет встроенную поддержку сокетов TCP

```
import socket
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect( ('data.pr4e.org', 80) )
```





http://docs.python.org/library/socket.html

https://xkcd.ru/i/353_v1.png

Прикладные протоколы

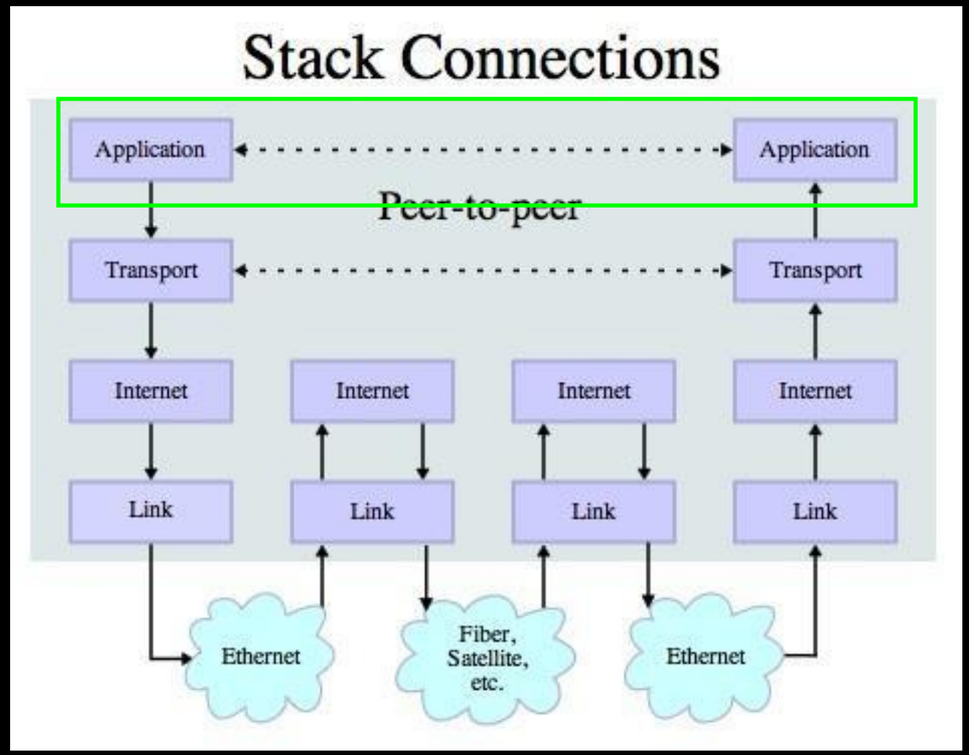
Прикладной протокол

- TCP (и Пайтон) предоставляет нам надежный сокет, но что мы хотим с этим сокетом делать? Какую проблему хотим решить?

- Прикладные протоколы:

  - Mail

  - World Wide Web



Источник: http://en.wikipedia.org/wiki/Internet_Protocol_Suite

HTTP – Протокол передачи гипертекста

- Доминирующий в Интернет протокол прикладного уровня передачи данных
- Был разработан для Веб, чтобы передавать HTML-страницы, картинки, документы, и т.д.
- Расширен для работы с данными в дополнение к документам: RSS, Web-сервисы и т.д. Базовая концепция: Создать соединение – Запросить документ – Получить документ – Закрыть соединение

https://ru.wikipedia.org/wiki/HTTP

HTTP
(Протокол передачи гипертекста)

Протокол Передачи Гипертекста — набор правил, позволяющих браузерам получать веб-документы с серверов через Интернет

Что такое протокол?

- Это свод правил, которым следуют все участники, чтобы можно было предсказать поведение каждого из участников

- Например, чтобы не столкнуться друг с другом:

  - На дорогах с двусторонним движением в США двигайтесь по правой стороне дороги

  - На дорогах с двусторонним движением в Великобритании двигайтесь по левой стороне дороги

`http://www.dr-chuck.com/page1.htm`

протокол хост документ

http://www.youtube.com/watch?v=x2GylLq59rI

1:17 - 2:19

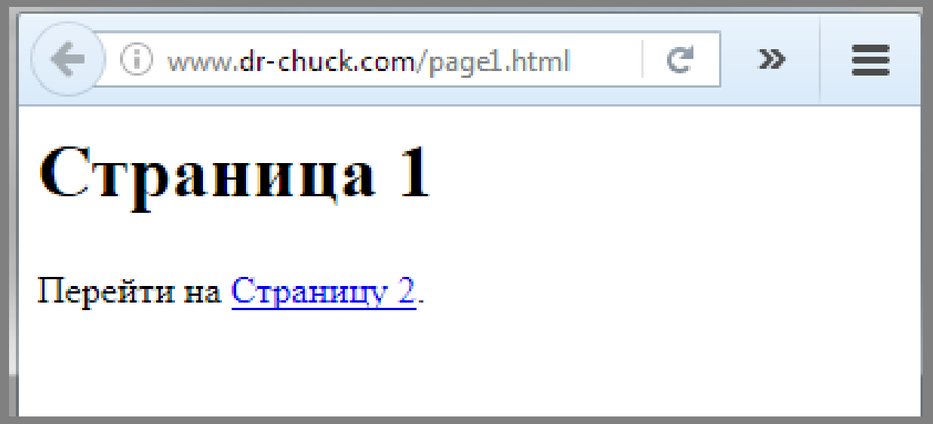
Получение данных с сервера

- Каждый раз, когда пользователь кликает на ссылку, имеющую атрибут href = value (в качестве значения – адрес документа, на который следует перейти), чтобы перейти на другую страницу, браузер устанавливает соединение с веб-сервером и делает запрос посредством метода “GET”, чтобы получить содержимое страницы по указанному адресу
- Сервер возвращает HTML-документ браузеру, который форматирует и отображает документ пользователю

Веб-сервер

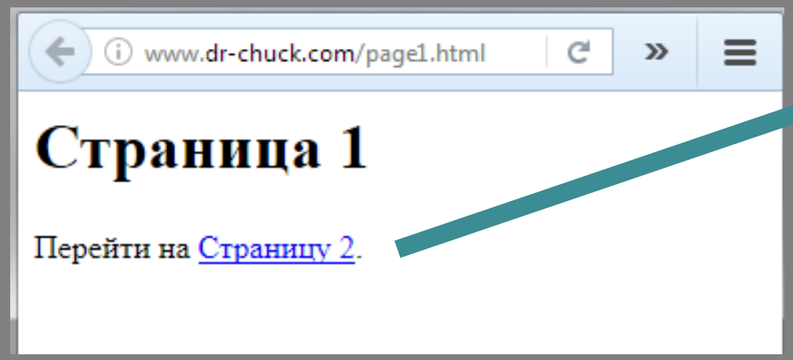
80

Браузер



Веб-сервер

80



Браузер

Клик по ссылке

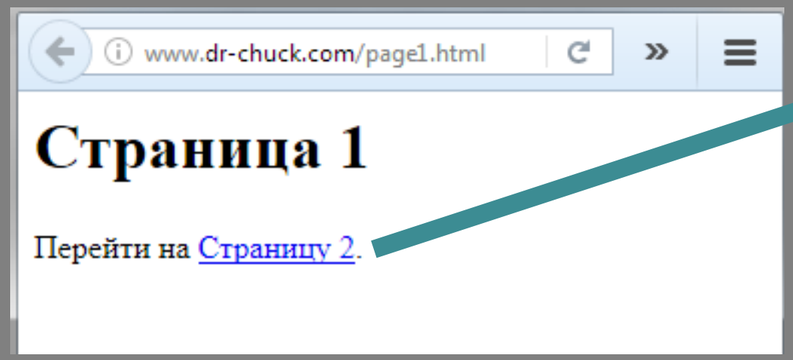
Запрос

Веб-сервер

80

GET
http://www.dr-chuck.com/page2.htm

Браузер



Клик по
ссылке

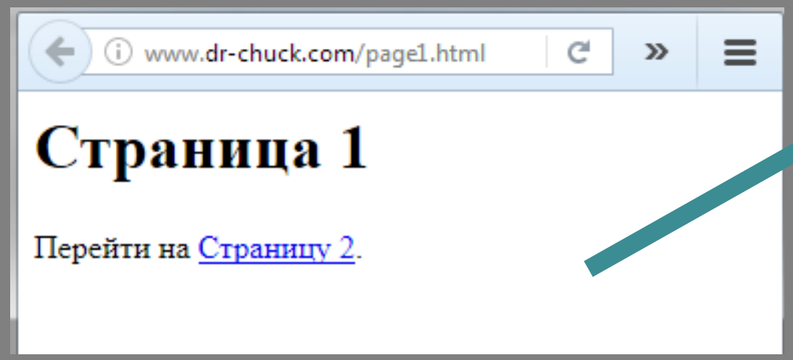
Запрос

Веб-сервер

80

GET
http://www.dr-chuck.com/page2.htm

Браузер



Клик по
ссылке

Запрос

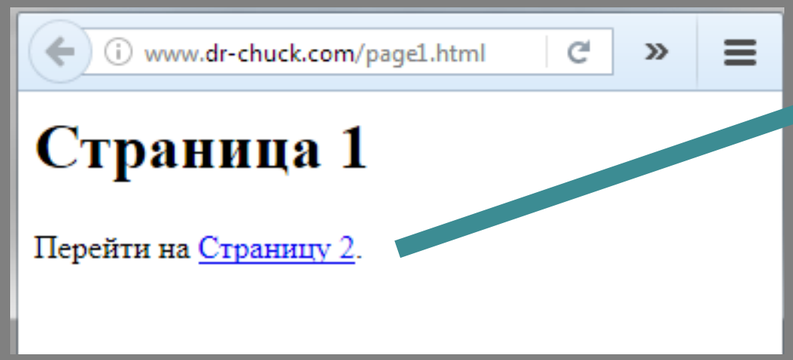
Веб-сервер

80

Ответ

<h1>Страница 2</h1>
Перейти на Страницу 1.

GET
http://www.dr-chuck.com/page2.htm



Браузер

Клик по ссылке

Запрос

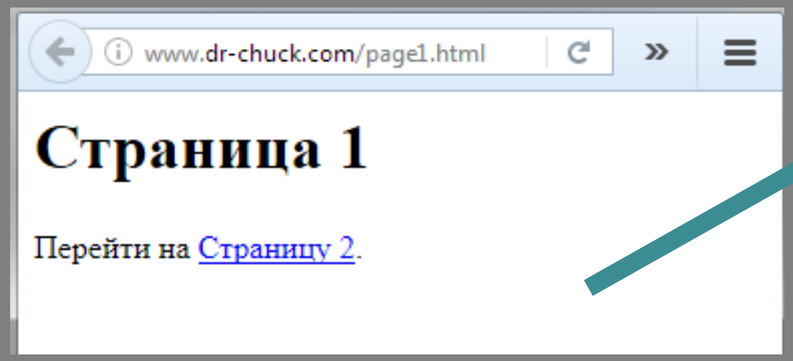
Веб-сервер

80

Ответ

<h1>Страница 2</h1>
Перейти на Страницу 1.

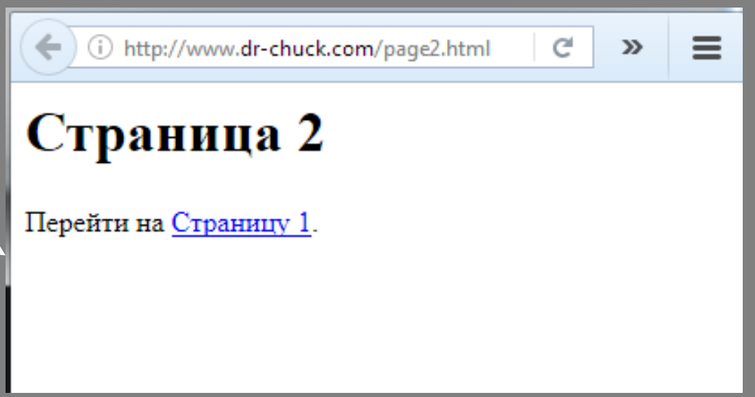
GET
http://www.dr-chuck.com/page2.htm



Браузер

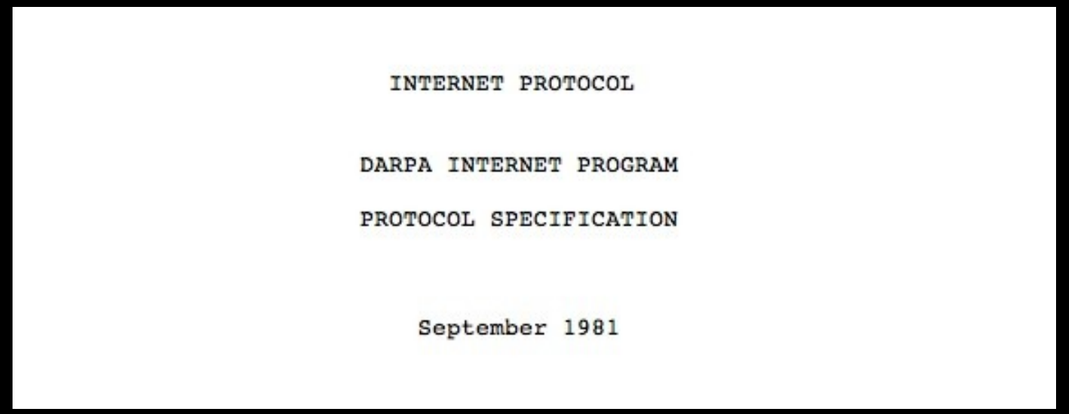
Клик по ссылке

Анализ/ Визуализация



Стандарты Интернета

- Стандарты для всех Интернет-протоколов (внутренние механизмы) разрабатываются организацией
- Internet Engineering Task Force (IETF) – Инженерный совет Интернета
- www.ietf.org
- Документ со стандартами называется «RFC» или «Рабочее предложение»



INTERNET PROTOCOL

DARPA INTERNET PROGRAM

PROTOCOL SPECIFICATION

September 1981

The internet protocol treats each internet datagram as an independent entity unrelated to any other internet datagram. There are no connections or logical circuits (virtual or otherwise).

The internet protocol uses four key mechanisms in providing its service: Type of Service, Time to Live, Options, and Header Checksum.

Источник: http://tools.ietf.org/html/rfc791

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

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

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information

http://www.w3.org/Protocols/rfc2616/rfc2616.txt

5 Request

A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.

```
Request       = Request-Line              ; Section 5.1
                *(( general-header        ; Section 4.5
                 | request-header         ; Section 5.3
                 | entity-header ) CRLF)  ; Section 7.1
                CRLF
                [ message-body ]          ; Section 4.3
```

5.1 Request-Line

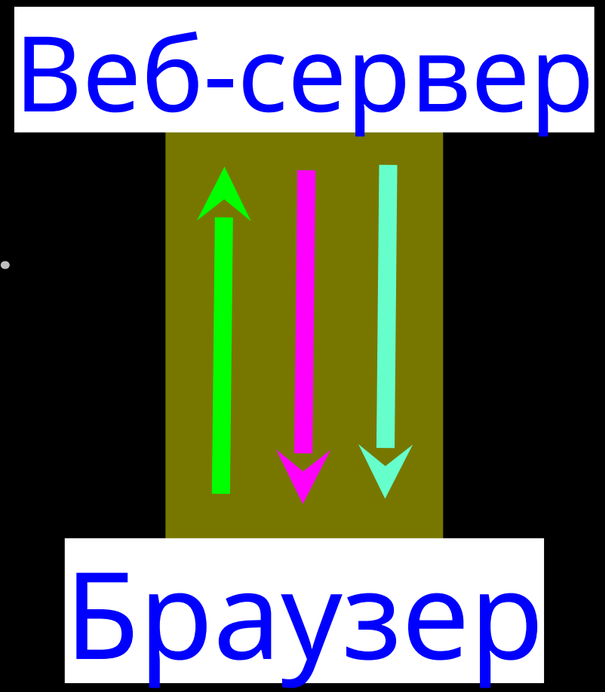
The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
Request-Line   = Method SP Request-URI SP HTTP-Version CRLF
```

Делаем HTTP-запрос

- Подключаемся к серверу, например, www.dr-chuck.com
- Запрашиваем документ (или документ по умолчанию)
    - GET http://www.dr-chuck.com/page1.htm HTTP/1.0
    - GET http://www.mlive.com/ann-arbor/ HTTP/1.0
    - GET http://www.facebook.com HTTP/1.0

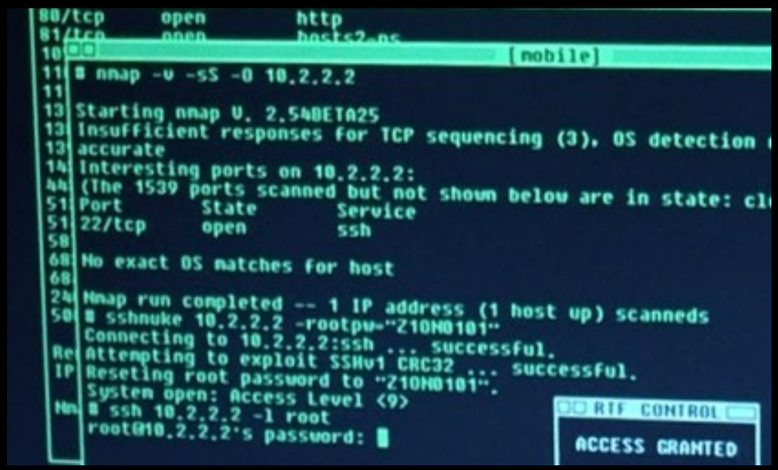
```
$ telnet www.dr-chuck.com 80
Trying 74.208.28.177...
Connected to www.dr-chuck.com.Escape character is '^]'.
GET http://www.dr-chuck.com/page1.htm HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 08 Jan 2015 01:57:52 GMT
Last-Modified: Sun, 19 Jan 2014 14:25:43 GMT
Connection: close
Content-Type: text/html

<h1>Страница 1</h1>
<p>Перейти на <a
href="http://www.dr-chuck.com/page2.htm">Страницу 2</a>.</p>
Соединение закрыто внешним хостом.
```



Меткий хакинг в кино

- Матрица Перезагрузка
- Ультиматум Борна
- Крепкий орешек 4
- ...

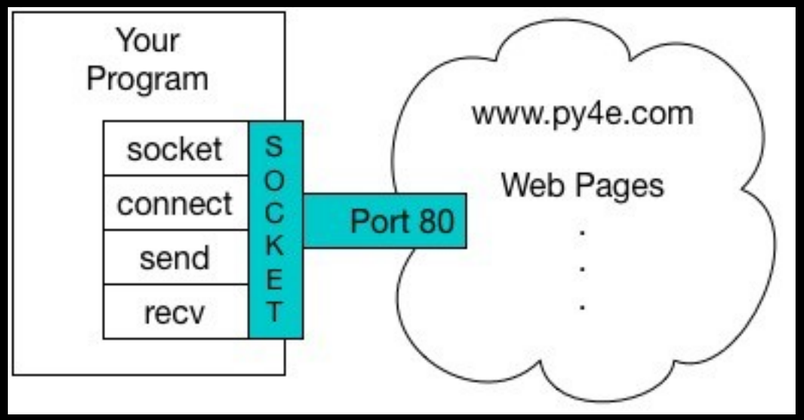
http://nmap.org/movies.html



Напишем веб-браузер!

HTTP-запрос в Пайтон

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('data.pr4e.org', 80))
cmd = 'GET http://data.pr4e.org/romeo.txt HTTP/1.0\r\n\r\n'.encode()
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if (len(data) < 1):
        break
    print(data.decode(),end='')
mysock.close()
```



```
HTTP/1.1 200 OK
Date: Sun, 14 Mar 2010 23:52:41 GMT
Server: Apache
Last-Modified: Tue, 29 Dec 2009 01:31:22 GMT
ETag: "143c1b33-a7-4b395bea"
Accept-Ranges: bytes
Content-Length: 167
Connection: close
Content-Type: text/plain
```

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief

Заголовок HTTP-запроса

```
while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) :
        break
    print(data.decode())
```

Тело HTTP-запроса

Символы и строки

ASCII

American Standard Code for Information Interchange

Американский стандартный код для обмена информацией

Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char
0	0x00	000	0000000	NUL	32	0x20	040	0100000	space	64	0x40	100	1000000	@	96	0x60	140	1100000	`
1	0x01	001	0000001	SOH	33	0x21	041	0100001	!	65	0x41	101	1000001	A	97	0x61	141	1100001	a
2	0x02	002	0000010	STX	34	0x22	042	0100010	"	66	0x42	102	1000010	B	98	0x62	142	1100010	b
3	0x03	003	0000011	ETX	35	0x23	043	0100011	#	67	0x43	103	1000011	C	99	0x63	143	1100011	c
4	0x04	004	0000100	EOT	36	0x24	044	0100100	$	68	0x44	104	1000100	D	100	0x64	144	1100100	d
5	0x05	005	0000101	ENQ	37	0x25	045	0100101	%	69	0x45	105	1000101	E	101	0x65	145	1100101	e
6	0x06	006	0000110	ACK	38	0x26	046	0100110	&	70	0x46	106	1000110	F	102	0x66	146	1100110	f
7	0x07	007	0000111	BEL	39	0x27	047	0100111	'	71	0x47	107	1000111	G	103	0x67	147	1100111	g
8	0x08	010	0001000	BS	40	0x28	050	0101000	(	72	0x48	110	1001000	H	104	0x68	150	1101000	h
9	0x09	011	0001001	TAB	41	0x29	051	0101001	)	73	0x49	111	1001001	I	105	0x69	151	1101001	i
10	0x0A	012	0001010	LF	42	0x2A	052	0101010	*	74	0x4A	112	1001010	J	106	0x6A	152	1101010	j
11	0x0B	013	0001011	VT	43	0x2B	053	0101011	+	75	0x4B	113	1001011	K	107	0x6B	153	1101011	k
12	0x0C	014	0001100	FF	44	0x2C	054	0101100	,	76	0x4C	114	1001100	L	108	0x6C	154	1101100	l
13	0x0D	015	0001101	CR	45	0x2D	055	0101101	-	77	0x4D	115	1001101	M	109	0x6D	155	1101101	m
14	0x0E	016	0001110	SO	46	0x2E	056	0101110	.	78	0x4E	116	1001110	N	110	0x6E	156	1101110	n
15	0x0F	017	0001111	SI	47	0x2F	057	0101111	/	79	0x4F	117	1001111	O	111	0x6F	157	1101111	o
16	0x10	020	0010000	DLE	48	0x30	060	0110000	0	80	0x50	120	1010000	P	112	0x70	160	1110000	p
17	0x11	021	0010001	DC1	49	0x31	061	0110001	1	81	0x51	121	1010001	Q	113	0x71	161	1110001	q
18	0x12	022	0010010	DC2	50	0x32	062	0110010	2	82	0x52	122	1010010	R	114	0x72	162	1110010	r
19	0x13	023	0010011	DC3	51	0x33	063	0110011	3	83	0x53	123	1010011	S	115	0x73	163	1110011	s
20	0x14	024	0010100	DC4	52	0x34	064	0110100	4	84	0x54	124	1010100	T	116	0x74	164	1110100	t
21	0x15	025	0010101	NAK	53	0x35	065	0110101	5	85	0x55	125	1010101	U	117	0x75	165	1110101	u
22	0x16	026	0010110	SYN	54	0x36	066	0110110	6	86	0x56	126	1010110	V	118	0x76	166	1110110	v
23	0x17	027	0010111	ETB	55	0x37	067	0110111	7	87	0x57	127	1010111	W	119	0x77	167	1110111	w
24	0x18	030	0011000	CAN	56	0x38	070	0111000	8	88	0x58	130	1011000	X	120	0x78	170	1111000	x
25	0x19	031	0011001	EM	57	0x39	071	0111001	9	89	0x59	131	1011001	Y	121	0x79	171	1111001	y
26	0x1A	032	0011010	SUB	58	0x3A	072	0111010	:	90	0x5A	132	1011010	Z	122	0x7A	172	1111010	z
27	0x1B	033	0011011	ESC	59	0x3B	073	0111011	;	91	0x5B	133	1011011	[	123	0x7B	173	1111011	{
28	0x1C	034	0011100	FS	60	0x3C	074	0111100	<	92	0x5C	134	1011100	\	124	0x7C	174	1111100	|
29	0x1D	035	0011101	GS	61	0x3D	075	0111101	=	93	0x5D	135	1011101	]	125	0x7D	175	1111101	}
30	0x1E	036	0011110	RS	62	0x3E	076	0111110	>	94	0x5E	136	1011110	^	126	0x7E	176	1111110	~
31	0x1F	037	0011111	US	63	0x3F	077	0111111	?	95	0x5F	137	1011111	_	127	0x7F	177	1111111	DEL

https://ru.wikipedia.org/wiki/ASCII
http://www.catonmat.net/download/ascii-cheat-sheet.png

Представление простых строк

- Каждый символ представлен числом от 0 до 256, хранящимся в 8 битах памяти
- Мы называем 8 бит памяти «байтом» (например, мой диск содержит 3 Терабайта памяти)
- Функция ord() сообщает нам числовое значение простого символа из таблицы ASCII

```
>>> print(ord('H'))
72
>>> print(ord('e'))
101
>>> print(ord('\n'))
10
>>>
```

ASCII

```
>>> print(ord('H'))
72
>>> print(ord('e'))
101
>>> print(ord('\n'))
10
>>>
```

В 1960-х и 1970-х считалось, что один байт – это один символ

Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char
0	0x00	000	0000000	NUL	32	0x20	040	0100000	space	64	0x40	100	1000000	@	96	0x60	140	1100000	`
1	0x01	001	0000001	SOH	33	0x21	041	0100001	!	65	0x41	101	1000001	A	97	0x61	141	1100001	a
2	0x02	002	0000010	STX	34	0x22	042	0100010	"	66	0x42	102	1000010	B	98	0x62	142	1100010	b
3	0x03	003	0000011	ETX	35	0x23	043	0100011	#	67	0x43	103	1000011	C	99	0x63	143	1100011	c
4	0x04	004	0000100	EOT	36	0x24	044	0100100	$	68	0x44	104	1000100	D	100	0x64	144	1100100	d
5	0x05	005	0000101	ENQ	37	0x25	045	0100101	%	69	0x45	105	1000101	E	101	0x65	145	1100101	e
6	0x06	006	0000110	ACK	38	0x26	046	0100110	&	70	0x46	106	1000110	F	102	0x66	146	1100110	f
7	0x07	007	0000111	BEL	39	0x27	047	0100111	'	71	0x47	107	1000111	G	103	0x67	147	1100111	g
8	0x08	010	0001000	BS	40	0x28	050	0101000	(	72	0x48	110	1001000	H	104	0x68	150	1101000	h
9	0x09	011	0001001	TAB	41	0x29	051	0101001	)	73	0x49	111	1001001	I	105	0x69	151	1101001	i
10	0x0A	012	0001010	LF	42	0x2A	052	0101010	*	74	0x4A	112	1001010	J	106	0x6A	152	1101010	j
11	0x0B	013	0001011	VT	43	0x2B	053	0101011	+	75	0x4B	113	1001011	K	107	0x6B	153	1101011	k
12	0x0C	014	0001100	FF	44	0x2C	054	0101100	,	76	0x4C	114	1001100	L	108	0x6C	154	1101100	l
13	0x0D	015	0001101	CR	45	0x2D	055	0101101	-	77	0x4D	115	1001101	M	109	0x6D	155	1101101	m
14	0x0E	016	0001110	SO	46	0x2E	056	0101110	.	78	0x4E	116	1001110	N	110	0x6E	156	1101110	n
15	0x0F	017	0001111	SI	47	0x2F	057	0101111	/	79	0x4F	117	1001111	O	111	0x6F	157	1101111	o
16	0x10	020	0010000	DLE	48	0x30	060	0110000	0	80	0x50	120	1010000	P	112	0x70	160	1110000	p
17	0x11	021	0010001	DC1	49	0x31	061	0110001	1	81	0x51	121	1010001	Q	113	0x71	161	1110001	q
18	0x12	022	0010010	DC2	50	0x32	062	0110010	2	82	0x52	122	1010010	R	114	0x72	162	1110010	r
19	0x13	023	0010011	DC3	51	0x33	063	0110011	3	83	0x53	123	1010011	S	115	0x73	163	1110011	s
20	0x14	024	0010100	DC4	52	0x34	064	0110100	4	84	0x54	124	1010100	T	116	0x74	164	1110100	t
21	0x15	025	0010101	NAK	53	0x35	065	0110101	5	85	0x55	125	1010101	U	117	0x75	165	1110101	u
22	0x16	026	0010110	SYN	54	0x36	066	0110110	6	86	0x56	126	1010110	V	118	0x76	166	1110110	v
23	0x17	027	0010111	ETB	55	0x37	067	0110111	7	87	0x57	127	1010111	W	119	0x77	167	1110111	w
24	0x18	030	0011000	CAN	56	0x38	070	0111000	8	88	0x58	130	1011000	X	120	0x78	170	1111000	x
25	0x19	031	0011001	EM	57	0x39	071	0111001	9	89	0x59	131	1011001	Y	121	0x79	171	1111001	y
26	0x1A	032	0011010	SUB	58	0x3A	072	0111010	:	90	0x5A	132	1011010	Z	122	0x7A	172	1111010	z
27	0x1B	033	0011011	ESC	59	0x3B	073	0111011	;	91	0x5B	133	1011011	[	123	0x7B	173	1111011	{
28	0x1C	034	0011100	FS	60	0x3C	074	0111100	<	92	0x5C	134	1011100	\	124	0x7C	174	1111100	|
29	0x1D	035	0011101	GS	61	0x3D	075	0111101	=	93	0x5D	135	1011101	]	125	0x7D	175	1111101	}
30	0x1E	036	0011110	RS	62	0x3E	076	0111110	>	94	0x5E	136	1011110	^	126	0x7E	176	1111110	~
31	0x1F	037	0011111	US	63	0x3F	077	0111111	?	95	0x5F	137	1011111	_	127	0x7F	177	1111111	DEL

Code Charts

Home | Site Map | Search

Unicode 9.0 Character Code Charts

SCRIPTS | SYMBOLS | NOTES

http://unicode.org/charts/

Find chart by hex code: Go Related links: Name index Help & links

Scripts

European Scripts	African Scripts	South Asian Scripts	Indonesia & Oceania Scripts
Armenian	*Adlam*	*Ahom*	*Balinese*
Armenian Ligatures	*Bamum*	*Bengali and Assamese*	*Batak*
Caucasian Albanian	Bamum Supplement	*Bhaiksuki*	*Buginese*
Cypriot Syllabary	*Bassa Vah*	*Brahmi*	*Buhid*
Cyrillic	*Coptic*	*Chakma*	*Hanunoo*
Cyrillic Supplement	*Coptic in Greek block*	*Devanagari*	*Javanese*
Cyrillic Extended-A	Coptic Epact Numbers	Devanagari Extended	*Rejang*
Cyrillic Extended-B	*Egyptian Hieroglyphs (1MB)*	*Grantha*	*Sundanese*
Cyrillic Extended-C	*Ethiopic*	*Gujarati*	Sundanese Supplement
Elbasan	Ethiopic Supplement	*Gurmukhi*	*Tagalog*
Georgian	Ethiopic Extended	*Kaithi*	*Tagbanwa*
Georgian Supplement	Ethiopic Extended-A	*Kannada*	**East Asian Scripts**
Glagolitic	*Mende Kikakui*	*Kharoshthi*	*Bopomofo*
Glagolitic Supplement	*Meroitic*	*Khojki*	Bopomofo Extended
Gothic	Meroitic Cursive	*Khudawadi*	*CJK Unified Ideographs (Han) (35MB)*
Greek	Meroitic Hieroglyphs	*Lepcha*	CJK Extension-A (6MB)
Greek Extended	*N'Ko*	*Limbu*	CJK Extension B (40MB)
Ancient Greek Numbers	*Osmanya*	*Mahajani*	CJK Extension C (3MB)
Latin	*Tifinagh*	*Malayalam*	CJK Extension D
Basic Latin (ASCII)	*Vai*	*Meetei Mayek*	CJK Extension E (3.5MB)
Latin-1 Supplement	**Middle Eastern Scripts**	Meetei Mayek Extensions	(see also Unihan Database)
Latin Extended-A	*Anatolian Hieroglyphs*	*Modi*	*CJK Compatibility Ideographs*

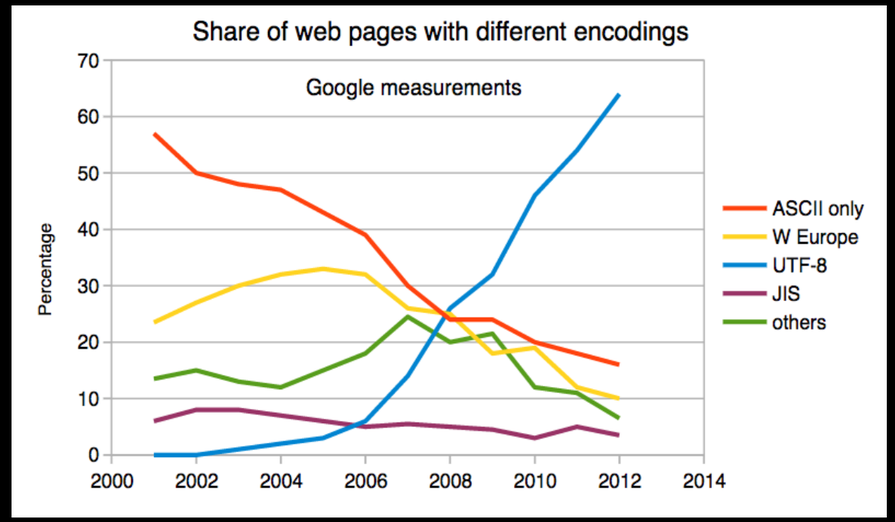
Многобайтовые символы

В связи с многообразием символов, которые должны обрабатываться компьютерами, существуют символы, представленные более чем одним байтом

UTF-16: фиксированная длина — два байта

- UTF-32: фиксированная длина — четыре байта
- UTF-8: 1-4 байта:
  - Полная обратная совместимость с ASCII;
  - Автоматическое определение между ASCII и UTF-8;
  - UTF-8 рекомендуется для кодирования данных, которыми обмениваются системы.

https://ru.wikipedia.org/wiki/UTF-8



Два вида строк в Пайтон

```
Python 2.7.10
>>> x = '이광춘'
>>> type(x)
<type 'str'>
>>> x = u'이광춘'
>>> type(x)
<type 'unicode'>
>>>
```

```
Python 3.5.1
>>> x = '이광춘'
>>> type(x)
<class 'str'>
>>> x = u'이광춘'
>>> type(x)
<class 'str'>
>>>
```

В Пайтон версии 3 все строки в кодировке Юникод

Пайтон 2 vs. Пайтон 3

```
Python 2.7.10
>>> x = b'abc'
>>> type(x)
<type 'str'>
>>> x = '이광춘'
>>> type(x)
<type 'str'>
>>> x = u'이광춘'
>>> type(x)
<type 'unicode'>
```

```
Python 3.5.1
>>> x = b'abc'
>>> type(x)
<class 'bytes'>
>>> x = '이광춘'
>>> type(x)
<class 'str'>
>>> x = u'이광춘'
>>> type(x)
<class 'str'>
```

Пайтон 3 и Юникод

- В Пайтон 3 все строки в кодировке Юникод
- Работа со строчными переменными в Пайтон, а также чтение данных из файла обычно не представляет сложности
- Но когда мы обращаемся к сетевому ресурсу, используя сокет, или обращаемся к базе данных необходимо кодировать и декодировать данные (обычно в UTF-8)

```
Python 3.5.1
>>> x = b'abc'
>>> type(x)
<class 'bytes'>
>>> x = '이광춘'
>>> type(x)
<class 'str'>
>>> x = u'이광춘'
>>> type(x)
<class 'str'>
```

Перевод строк в байты в Пайтон

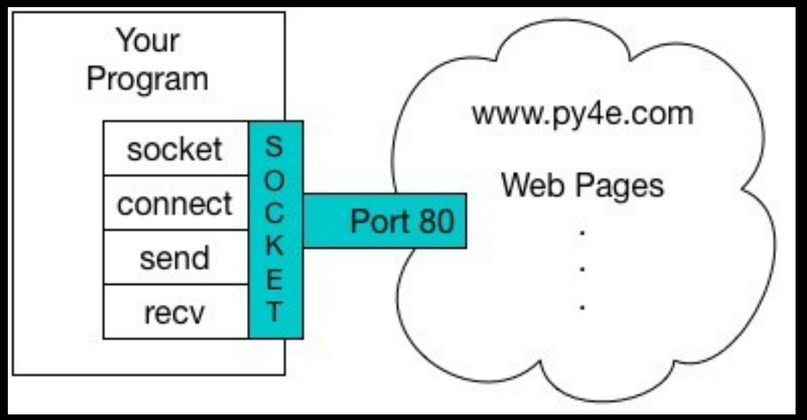
- Когда мы обращаемся к внешнему ресурсу вроде сетевого сокета, то отправляем байты. Поэтому необходимо перевести строки Пайтон 3 в заданную кодировку

- Когда мы читаем данные с внешнего ресурса, их нужно декодировать в набор символов, чтобы в Пайтон 3 они корректно преобразовались в строку:

```
while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) :
        break
    mystring = data.decode()
    print(mystring)
```

HTTP-запрос в Пайтон

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('data.pr4e.org', 80))
cmd = 'GET http://data.pr4e.org/romeo.txt HTTP/1.0\n\n'.encode()
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if (len(data) < 1):
        break
    print(data.decode())
mysock.close()
```



`bytes.`**`decode`**`(`*`encoding="utf-8", errors="strict"`*`)`

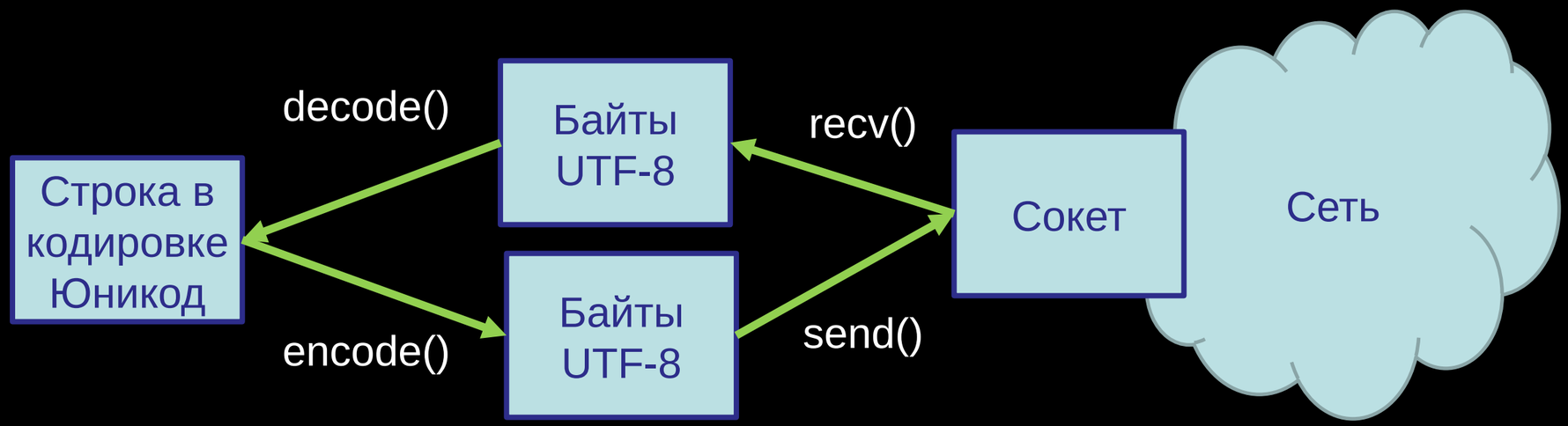
`bytearray.`**`decode`**`(`*`encoding="utf-8", errors="strict"`*`)`

Return a string decoded from the given bytes. Default encoding is `'utf-8'`. *errors* may be given to set a different error handling scheme. The default for *errors* is `'strict'`, meaning that encoding errors raise a `UnicodeError`. Other possible values are `'ignore'`, `'replace'` and any other name registered via `codecs.register_error()`, see section Error Handlers. For a list of possible encodings, see section Standard Encodings.

`str.`**`encode`**`(`*`encoding="utf-8", errors="strict"`*`)`

Return an encoded version of the string as a bytes object. Default encoding is `'utf-8'`. *errors* may be given to set a different error handling scheme. The default for *errors* is `'strict'`, meaning that encoding errors raise a `UnicodeError`. Other possible values are `'ignore'`, `'replace'`, `'xmlcharrefreplace'`, `'backslashreplace'` and any other name registered via `codecs.register_error()`, see section Error Handlers. For a list of possible encodings, see section Standard Encodings.

https://docs.python.org/3/library/stdtypes.html#bytes.decode
https://docs.python.org/3/library/stdtypes.html#str.encode




```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('data.pr4e.org', 80))
cmd = 'GET http://data.pr4e.org/romeo.txt HTTP/1.0\n\n'.encode()
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if (len(data) < 1):
        break
    print(data.decode())
mysock.close()
```

Упрощение HTTP с помощью модуля urllib

Использование urllib в Пайтон

Благодаря распространенности HTTP, у нас есть библиотека, которая производит всю работу с сокетами и делает веб-страницы похожими на файл

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
for line in fhand:
    print(line.decode().strip())
```

urllib1.py

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')
for line in fhand:
    print(line.decode().strip())
```

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief

urllib1.py

Похоже на файл...

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://data.pr4e.org/romeo.txt')

counts = dict()
for line in fhand:
    words = line.decode().split()
    for word in words:
        counts[word] = counts.get(word, 0) + 1
print(counts)
```

urlwords.py

Чтение веб-страниц

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print(line.decode().strip())
```

```
<h1>Страница 1</h1>
<p>Перейти на<a
href="http://www.dr-chuck.com/page2.htm">Страни
цу 2</a>.
</p>
```

urllib2.py

Следуем по ссылкам

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print(line.decode().strip())
```

```
<h1>Страница 1</h1>
<p>Перейти на <a href="http://www.dr-
chuck.com/page2.htm">Страницу 2</a>.
</p>
```

urllib2.py

Первые строки кода @ Google?

```
import urllib.request, urllib.parse, urllib.error

fhand = urllib.request.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print(line.decode().strip())
```

HTML-парсинг
(или веб-скрейпинг)

Что такое веб-скрейпинг?

- Программа или скрипт, притворяясь браузером, извлекает и анализирует информацию с веб-страниц, а затем отправляется по ссылкам на следующие страницы
- Поисковый робот, обрабатывающий веб-страницы, называется «веб-паук» или «веб-краулер»

https://ru.wikipedia.org/wiki/Веб-скрейпинг
https://ru.wikipedia.org/wiki/Поисковый_робот

Зачем нужен веб-скрейпинг?

- Извлечь данные, особенно из соц.сетей: кто на кого ссылается?
- Получить обратно свои данные из системы, не имеющей возможности экспорта данных
- Мониторить сайт на предмет новой информации
- Создать базу данных для поискового робота

Скрейпинг веб-страниц

- Все еще существуют разногласия по поводу использования веб-скрейпинга, и некоторые сайты особенно непримиримы в этом отношении
- Не разрешается повторная публикация информации, защищенной авторским правом
- Не допускается нарушение условий пользования сервисом

Простой способ — Beautiful Soup

- можно выполнять поиск по строкам сложным путем
- или использовать бесплатную библиотеку BeautifulSoup с сайта www.crummy.com

https://www.crummy.com/software/BeautifulSoup/

Установка BeautifulSoup

```
# Вы можете установить BeautifulSoup, перейдя по ссылке:
# https://pypi.python.org/pypi/beautifulsoup4

# Или скачать файл:
# http://www.py4e.com/code3/bs4.zip
# и распаковать его в ту же директорию, что и этот файл

import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup

...
```

urllinks.py

```
import urllib.request, urllib.parse,
urllib.error
from bs4 import BeautifulSoup

url = input('Enter - ')
html = urllib.request.urlopen(url).read()
soup = BeautifulSoup(html, 'html.parser')

# Получить все теги типа «ссылка»
tags = soup('a')
for tag in tags:
    print(tag.get('href', None))
```

```
python urllinks.py
Enter - http://www.dr-chuck.com/page1.htm
http://www.dr-chuck.com/page2.htm
```

Резюме

- TCP/IP создает для нас каналы / сокеты между приложениями
- Для использования этих каналов разработаны прикладные протоколы
- Протокол передачи гипертекста (HTTP) — простой, но мощный протокол
- Пайтон имеет хорошую поддержку сокетов, HTTP и HTML-парсинга

Авторы / Благодарности

… Insert new Contributors and Translations here

Исходная разработка: Чарльз Северанс, Школа Информации Мичиганского Университета.

Перевод выполнила Фомкина Виолетта.

… Добавьте сюда новых авторов и переводчиков

...