Стандартная библиотека: сокеты

- socket - низкоуровневая работа с сетью

```
1     def client(host, port=12000):
2         # tcp сокет
3         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
4         # socket.SOCK_DGRAM – udp
5         sock.connect((host, port))
6         sock.send("HELLO")
7         print sock.recv()
8
9     def server(host='0.0.0.0', port=12000):
10        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11        sock.bind((host, port))
12        sock.listen(5)
13
14        while True:
15            cs, client_addr_info = sock.accept()
16            print "connection␣from␣", client_addr_info
17            cs.recv(5)
18            cs.send("BYE!")
19            cs.close()
```

Стандартная библиотека: сокеты

- gethostbyname, socket.gethostbyname_ex - DNS запрос
- gethostbyaddr - reverse DNS
- setdefaulttimeout(timeout) - установка таймаута для операциях на сокетах
- htonl - преобразование порядка байтов,.....
- ssl - ssl сокеты

```
1  import socket
2
3  print socket.gethostbyaddr('8.8.8.8')
4  # ('google-public-dns-a.google.com', [], ['8.8.8.8'])
5
6  print socket.gethostbyname_ex("www.google.com")
7  # ('www.l.google.com', ['www.google.com'],
8  #   ['173.194.35.145', '173.194.35.146', '173.194.35.147',
9  #            '173.194.35.148', '173.194.35.144'])
```

Стандартная библиотека: сокеты

- select - асинхронная работа с сокетами
- select.select(read_list, write_list, error_list, timeout=None)
- select.pool, select.epool - быстрее на больших списках дескрипторов

```
1   import select
2
3   sc1 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
4   sc2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5   sc1.connect(("www.google.com", 80))
6   sc2.connect(("www.google.com", 80))
7
8   r,w,e = select.select([sc1, sc2], [], [], 0.1)
9
10  r == []
11  w == []
12  e == []
```

Другие сетевые библиотеки

- scapy - ICMP, ARP, манипуляция сетевыми пакетами, etc
- twisted - асинхроннай фреймворк, поддерживает большую часть используемых сетевых протоколов
- gevent - асинхронный фреймворк, поддерживает программирование без callback, эмулирует потоки

```
1   from twisted.web import server, resource
2   from twisted.internet import reactor
3
4   class HelloResource(resource.Resource):
5       isLeaf = True
6       numberRequests = 0
7
8       def render_GET(self, request):
9           self.numberRequests += 1
10          request.setHeader("content-type", "text/plain")
11          return "I am request #" + str(self.numberRequests) +
12
13  reactor.listenTCP(8080, server.Site(HelloResource()))
14  reactor.run()
```

HTTP

- Стандартная библиотека: httplib, urllib, urllib2

```
1       import urllib2
2       url = "http://search.yahoo.com/search?p=test"
3       search_res = urllib2.urlopen(url).read()
```

- urllib3
- requests

Стандартная библиотека: Другие протоколы

- poplib
- imaplib
- ftplib
- telnetlib
- BaseHttpServer
- ...