

Sockets

`s = socket.socket(socket_family, socket_type, protocol=0)`

`socket_family`: This is either `AF_UNIX` or `AF_INET`, as explained earlier.

`socket_type`: This is either `SOCK_STREAM` or `SOCK_DGRAM`.

`AF_INET`: TCPv4 or Hostname

`SOCK_DGRAM`: UDP

`SOCK_STREAM`: `SOCK_STREAM` or `SOCK_DGRAM`.

`SOCK_DGRAM`: TCP

`protocol`: This is usually left out, defaulting to 0.

Domain The family of protocols that is used as the transport mechanism. These values are constants such as `AF_INET`, `PF_INET`, `PF_UNIX`, `PF_X25`, and so on.

Type The type of communications between the two endpoints, typically `SOCK_STREAM` for connection-oriented protocols and `SOCK_DGRAM` for connectionless protocols.

Sockets (cont)

protocol Typically zero, this may be used to identify a variant of a protocol within a domain and type.

hostname The identifier of a network interface:

A string, which can be a host name, a dotted-quad address, or an IPV6 address in colon (and possibly dot) notation

A string "<broadcast>", which specifies an `INADDR_BROADCAST` address.

A zero-length string, which specifies `INADDR_ANY`, or An Integer, interpreted as a binary address in host byte order.

Server Socket Methods

`s.bind()` This method binds address (hostname, port number pair) to socket.

`s.listen()` This method sets up and start TCP listener.

`s.accept()` This passively accept TCP client connection, waiting until connection arrives (blocking).

Client Socket Method

`s.connect()` This method actively initiates TCP server connection.

General Socket Method

`s.recv()` This method receives TCP message

`s.send()` This method transmits TCP message

`s.recvfrom()` This method receives UDP message

`s.sendto()` This method transmits UDP message

`s.close()` This method closes socket

`socket.gethostname()` Returns the hostname.

Simple Server Example

```
#!/usr/bin/python # This is
server.py file
import socket # Import socket
module
s = socket.socket() # Create a
socket object
host = socket.gethostname() # Get
local machine name
port = 12345 # Reserve a port for
your service.
s.bind((host, port)) # Bind to the
port
s.listen(5) # Now wait for client
connection.
while True:
```

Simple Server Example (cont)

```
c, addr = s.accept() # Establish connection with
client.
print 'Got connection from', addr
c.send('Thank you for connecting')
c.close() # Close the connection
```

Simple Client Example

```
#!/usr/bin/python # This is client.py file
import socket # Import socket module
s = socket.socket() # Create a socket object
host = socket.gethostname() # Get local machine
name
port = 12345 # Reserve a port for your service.
s.connect((host, port))
print s.recv(1024)
s.close # Close the socket when done
```



By **k0ncept**
cheatography.com/k0ncept/

Not published yet.
Last updated 25th August, 2016.
Page 2 of 2.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>