**AMALIY MASHG‘ULOT UCHUN O‘QUV MATERIALLARI**

**3-Mavzu:** Pythonda tarmoq dasturlashga kirish.

**1-mashg‘ulot.** Tarmoqda ma’lumot almashuvchi klient-server dasturini tuzish.

**O‘quv savollari:**

1. Socket modulining asosiy metodlari bilan tanishish.

**1. Socket modulining asosiy metodlari bilan tanishish**

Python tarmoq xizmatlariga kirishning ikki darajasini ta'minlaydi. Past darajada, siz asosiy operatsion tizimdagi asosiy rozetka yordamiga kirishingiz mumkin, bu sizga ulanishga yo‘naltirilgan va ulanishsiz protokollar uchun mijozlar va serverlarni amalga oshirish imkonini beradi.

Python, shuningdek, FTP, HTTP va boshqalar kabi ma'lum amaliy qatlam tarmoq protokollariga yuqori darajadagi kirishni ta'minlaydigan kutubxonalarga ega.

Ushbu bo‘lim sizga Internetdagi eng mashhur kontseptsiya - Socket Programming haqida tushuncha beradi.

**Soketlar nima?**

Soketlar ikki tomonlama aloqa kanalining so‘nggi nuqtalari hisoblanadi. Soketlar jarayon ichida, bitta mashinadagi jarayonlar o‘rtasida yoki turli qit'alardagi jarayonlar o‘rtasida aloqa qilishi mumkin.

Soketlar bir nechta turli turdagi quvurlar uchun amalga oshirilishi mumkin: Unix domen soketlari, TCP, UDP va boshqalar. *Soket* kutubxonasi umumiy transportlarni boshqarish uchun maxsus sinflarni, shuningdek, qolganlarini boshqarish uchun umumiy interfeysni taqdim etadi.

Soketlarning o‘z lug‘ati bor.

|  |  |
| --- | --- |
| **t/r** | **Muddati va tavsifi** |
| 1 | **Domen**  Transport mexanizmi sifatida foydalaniladigan protokollar oilasi. Bu qiymatlar AF\_INET, PF\_INET, PF\_UNIX, PF\_X25 va boshqalar kabi konstantalardir. |
| 2 | **turi**  Ikki soʻnggi nuqta oʻrtasidagi aloqa turi, odatda ulanishga yoʻnaltirilgan protokollar uchun SOCK\_STREAM va ulanishsiz protokollar uchun SOCK\_DGRAM. |
| 3 | **protokol**  Odatda nolga teng, bu domen va turdagi protokol variantini aniqlash uchun ishlatilishi mumkin. |
| 4 | **xost nomi**  Tarmoq interfeysi identifikatori -   * Xost nomi, toʻrt nuqtali manzil yoki ikki nuqta (va ehtimol nuqta) belgisidagi IPV6 manzili boʻlishi mumkin boʻlgan qator. * INADDR\_BROADCAST manzilini bildiruvchi “<broadcast>” qatori. * INADDR\_ANY ni belgilaydigan nol uzunlikdagi qator yoki * Xost bayt tartibida ikkilik manzil sifatida talqin qilingan butun son. |
| 5 | **port**  Har bir server bir yoki bir nechta portga qo‘ng‘iroq qilayotgan mijozlarni tinglaydi. Port Fixnum port raqami, port raqami qatori yoki xizmat nomi bo‘lishi mumkin. |

**Domen**

Transport mexanizmi sifatida foydalaniladigan protokollar oilasi. Bu qiymatlar AF\_INET, PF\_INET, PF\_UNIX, PF\_X25 va boshqalar kabi konstantalardir.

**type**

Ikki soʻnggi nuqta oʻrtasidagi aloqa turi, odatda ulanishga yoʻnaltirilgan protokollar uchun SOCK\_STREAM va ulanishsiz protokollar uchun SOCK\_DGRAM.

**protokol**

Odatda nolga teng, bu domen va turdagi protokol variantini aniqlash uchun ishlatilishi mumkin.

**xost nomi**

Tarmoq interfeysi identifikatori -

Xost nomi, toʻrt nuqtali manzil yoki ikki nuqta (va ehtimol nuqta) belgisidagi IPV6 manzili boʻlishi mumkin boʻlgan qator.

INADDR\_BROADCAST manzilini bildiruvchi “<broadcast>” qatori.

INADDR\_ANY ni belgilaydigan nol uzunlikdagi qator yoki

Xost bayt tartibida ikkilik manzil sifatida talqin qilingan butun son.

**port**

Har bir server bir yoki bir nechta portga qo‘ng‘iroq qilayotgan mijozlarni tinglaydi. Port Fixnum port raqami, port raqami qatori yoki xizmat nomi bo‘lishi mumkin.

***soket* moduli**

Soket yaratish uchun umumiy sintaksisga ega *socket* modulida mavjud *socket.socket() funksiyasidan foydalanishingiz kerak.*

s = socket.socket(socket\_family, socket\_type, protocol=0)

Bu erda variantlarning tavsifi:

* **socket\_family** AF\_UNIX yoki AF\_INET, avvalroq tushuntirilganidek.
* **socket\_type** - SOCK\_STREAM yoki SOCK\_DGRAM.
* **protokol** - odatda ko‘rsatilmaydi, standart 0.

**socket\_family** AF\_UNIX yoki AF\_INET, avvalroq tushuntirilganidek.

**socket\_type** - SOCK\_STREAM yoki SOCK\_DGRAM.

**protokol** - odatda ko‘rsatilmaydi, standart 0.

*Soket* ob'ektiga ega bo‘lganingizdan so‘ng, mijoz yoki server dasturini yaratish uchun kerakli funktsiyalardan foydalanishingiz mumkin. Quyida talab qilinadigan xususiyatlar ro‘yxati -

**Server soket usullari**

|  |  |
| --- | --- |
| **t/r** | **Usul va tavsif** |
| 1 | **s.bind()**  Ushbu usul manzilni (xost nomi, port raqamlari juftligi) rozetkaga bog‘laydi. |
| 2 | **s.listen()**  Ushbu usul TCP tinglovchisini o‘rnatadi va ishga tushiradi. |
| 3 | **s.accept()**  Bu passiv ravishda TCP mijoz ulanishini qabul qiladi, ulanish o‘rnatilguncha kutadi (blokirovka). |

**s.bind()**

Ushbu usul manzilni (xost nomi, port raqamlari juftligi) rozetkaga bog‘laydi.

**s.tinglash()**

Ushbu usul TCP tinglovchisini o‘rnatadi va ishga tushiradi.

**s.accept()**

Bu passiv ravishda TCP mijoz ulanishini qabul qiladi, ulanish o‘rnatilguncha kutadi (blokirovka).

**Mijoz soket usullari**

|  |  |
| --- | --- |
| **t/r** | **Usul va tavsif** |
| 1 | **s.connect()**  Ushbu usul TCP serveriga ulanishni faol ravishda boshlaydi. |

**s.connect()**

Ushbu usul TCP serveriga ulanishni faol ravishda boshlaydi.

**Umumiy soket usullari**

|  |  |
| --- | --- |
| **t/r** | **Usul va tavsif** |
| 1 | **s.recv()**  Ushbu usul TCP xabarini oladi |
| 2 | **s.send()**  Ushbu usul TCP xabarini yuboradi |
| 3 | **s.recvfrom()**  Bu usul UDP xabarini oladi |
| 4 | **s.sendto()**  Bu usul UDP xabarini yuboradi |
| 5 | **s.close()**  Ushbu usul rozetkani yopadi |
| 6 | **socket.getostname()**  Xost nomini qaytaradi. |

**s.recv()**

Ushbu usul TCP xabarini oladi

**s.send()**

Ushbu usul TCP xabarini yuboradi

**s.recvfrom()**

Bu usul UDP xabarini oladi

**s.sendto()**

Bu usul UDP xabarini yuboradi

**s.close()**

Ushbu usul rozetkani yopadi

**socket.getostname()**

Xost nomini qaytaradi.

**oddiy server.** Internet-serverlarni yozish uchun biz rozetka ob'ektini yaratish uchun rozetka modulida mavjud **soket funksiyasidan foydalanamiz.**Soket ob'ekti soket serverini sozlash uchun boshqa funktsiyalarni chaqirish uchun ishlatiladi.

Endi berilgan hostda xizmatingiz uchun *portni* belgilash uchun **bog‘lash (hostname, port) funksiyasini chaqiring.**

Keyin qaytarilgan ob'ektni *qabul qilish usulini chaqiring.*Ushbu usul mijoz siz ko‘rsatgan portga ulanishini kutadi va keyin ushbu mijozga ulanishni ifodalovchi *Connection ob'ektini qaytaradi.*

#!/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:

c, addr = s.accept() # Establish connection with client.

print 'Got connection from', addr

c.send('Thank you for connecting‘)

c.close() # Close the connection

**Oddiy mijoz.** Berilgan port 12345 va berilgan xostga ulanishni ochadigan juda oddiy mijoz dasturini yozamiz. *Python soket* moduli funksiyasidan foydalangan holda soket mijozini yaratish juda oddiy .

**Socket.connect(hostname, port)** portdagi *xost nomi bilan* TCP ulanishini ochadi. Ochiq rozetkaga ega bo‘lganingizdan so‘ng, siz undan istalgan kiritish-chiqarish ob'ekti kabi o‘qishingiz mumkin. Ishingiz tugagach, faylni yopganingizdek, uni yopishni unutmang.

Quyidagi kod juda oddiy mijoz bo‘lib, u berilgan xost va portga ulanadi, rozetkadan barcha mavjud ma'lumotlarni o‘qiydi va keyin chiqadi.

#!/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

Endi bu server.py ni fonda ishga tushiring va natijani ko‘rish uchun yuqoridagi client.py ni ishga tushiring.

# Following would start a server in background.

$ python server.py &

# Once server is started run client as follows:

$ python client.py

Bu quyidagi natijani beradi:

Got connection from ('127.0.0.1', 48437)

Thank you for connecting

**Python Internet modullari**

Python Network/Internet dasturlashdagi ba'zi muhim modullar ro‘yxati.

|  |  |  |  |
| --- | --- | --- | --- |
| **protokol** | **Umumiy funktsiya** | **Port raqami.** | **Python moduli** |
| HTTP | veb-sahifalar | 80 | httplib, urllib, xmlrpclib |
| NNTP | Usenet yangiliklari | 119 | nntplib |
| FTP | Fayl uzatish | 20 | ftplib, urllib |
| SMTP | Email yuborish | 25 | smtplib |
| POP3 | Email qabul qilinmoqda | 110 | poplib |
| IMAP4 | Email qabul qilinmoqda | 143 | imaplib |
| telnet | Buyruqlar qatorlari | 23 | telnetlib |
| gofer | Hujjatlarni topshirish | 70 | gopher, urlib |

Iltimos, FTP, SMTP, POP va IMAP protokollari bilan ishlash uchun yuqorida ko‘rsatilgan barcha kutubxonalarni tekshiring.

**Nazorat savollari:**

1. Tarmoq dasturlash deganda nimani tushunasiz?
2. Tarmoqda ishlash uchun python tilining qaysi paketi bilan ishlanadi?
3. Qanday metodalarni bilasiz?
4. Bind() metodining vazifasi qanday?
5. connect metodining vazifasi qanbday?
6. Protokol nima?