< VOLTAR



Recebendo uma conexão em um servidor Python

Criar um exemplo de aplicação trocando mensagens entre uma conexão cliente e servidor.

NESTE TÓPICO

Dê uma olhada nos links abaixo para saber mais sobre a linguagem Python:

> Referências

Marcar tópico





Olá alunos.

Vamos ver um exemplo de aplicação, criando um servidor e um cliente Python e trocando mensagens entre eles.

O primeiro passo é criar o **servidor Python** através de um **script**, para isso vamos abrir o **IDLE** e em **FILE**, **NEW FILE** e digitar:

```
    import socket
    host = '127.0.0.1' # o mesmo que localhost
    porta = 8800 # porta da conexão
```

Como vamos usar o módulo **socket**, então importamos todas as funções deste módulo (linha 1). Depois criamos o endereço do **host** e da **porta** da conexão, no cliente deverá ter o mesmo valor para o **host** e para a **porta**. O IP 127.0.0.1 é o mesmo que localhost, pois se trata de uma conexão local, tipo intranet.

```
1. import socket
2.
3. host = '127.0.0.1' # o mesmo que localhost
4. porta = 8800 # porta da conexão
5.
6. soquete = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #estou usando TCP/IP
7. recebe = (host, porta)
8. soquete.bind(recebe)
9. soquete.listen(2)
```

Continuando...na linha **6**, criamos uma variável **soquete** e atribuindo, chamamos a função socket do módulo socket e mais duas funções: **AF_INET**: para conexão entre endereços IP e é a que vai determinar um valor para a porta e **SOCK_STREAM**: porque vamos usar o protocolo **TCP** (se fosse usar outro protocolo de transmissão como UDP, por exemplo, teríamos que usar: SOCK_DGRAM).

Na linha **7**, criamos a variável **recebe** e atribuímos os valores do **host** e da **porta**.

Na linha **8**, a variável **soquete** utiliza a função **bind**, para apanhar os valores da variável **recebe** (colocar um valor de IP e da porta no socket).

Na linha **9**, a conexão atribuída a variável **soquete** aceita (ouve) até duas conexões cliente, você pode aumentar este valor conforme a necessidade.

```
1. import socket
 2.
 3. host = '127.0.0.1' # o mesmo que localhost
 4. porta = 8800
                        # porta da conexão
 5.
    soquete = socket.socket(socket.AF INET, socket.SOCK STREAM) #estou usando TCP/IP
 6.
7. recebe = (host, porta)
 8.
    soquete.bind(recebe)
9.
    soquete.listen(2)
10.
11. print('\nSERVIDOR INICIADO...IP: ', host, 'PORTA: ', porta)
12.
13. while True:
       conexao, enderecoIP = soquete.accept()
14.
        print('\nSERVIDOR ACESSADO PELO CLIENTE:', enderecoIP)
15.
```

Na linha **11**, exibe uma mensagem quando subirmos o servidor, mostrando o endereço do IP do servidor e o número da **porta**.

Na linha 13, abrimos um loop com while, passando o valor booleano True, isto é equivalente ao comando em C: loop(), ou seja, todos os comandos abaixo dele serão repetidos até que ocorra alguma interrupção. Na linha 14, criamos as variáveis conexao e enderecolP recebendo e aceitando a conexão do cliente através da função accept().

Na linha **15**, a mensagem será exibida no servidor quando o cliente for executado, mostrando o endereço de **IP do cliente** e um valor para **porta**, gerada pela função **AF_INET**.

```
1. import socket
 2.
3. host = '127.0.0.1' # o mesmo que localhost
                        # porta da conexão
4. porta = 8800
 5.
6. soquete = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #estou usando TCP/IP
7. recebe = (host, porta)
8. soquete.bind(recebe)
soquete.listen(2)
10.
11. print('\nSERVIDOR INICIADO...IP: ', host, 'PORTA: ', porta)
12.
13. while True:
        conexao, enderecoIP = soquete.accept()
14.
        print('\nSERVIDOR ACESSADO PELO CLIENTE:', enderecoIP)
15.
16.
17.
        while True:
18.
            mensagem = conexao.recv(2048)
            if not mensagem:
19.
20.
                break
         print('\nIP CLIENTE:', enderecoIP)
21.
           print('MENSAGEN RECEBIDA: ', mensagem.decode())
22.
23.
24.
        print('CONEXÃO COM O CLIENTE FINALIZADA...', enderecoIP)
25.
        conexao.close()
```

Continuando...na linha 17, abrimos outro loop com while e True para receber a mensagem do cliente. Criamos a variável mensagem que recebe o valor da variável conexao que o soquete aceitou e através da função recv(2048), recebe o texto da mensagem, o valor entre parêntesis, 2048, significa que vai aceitar textos com até 2K Bytes de caracteres. Você pode alterar este valor.

Na linha **19**, temos um **if** que, senão receber mais nenhuma mensagem, o loop será interrompido.

Nas linhas **21 e 22**, no servidor será exibido o **número do IP** e a **mensagem recebida** do **cliente**.

Na linha **24**, será exibida a mensagem caso o **cliente** interromper a conexão.

Antes de finalizarmos o script, vamos incrementar mais um pouco o código:

```
1. # CRIANDO UM SERVIDOR PYTHON
 import socket
3. from datetime import *
 4.
 5. host = '127.0.0.1' # o mesmo que localhost
6. porta = 8800
                        # porta da conexão
7.
8. soquete = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #estou usando TCP/IP
9. recebe = (host, porta)
10. soquete.bind(recebe)
11. soquete.listen(2)
12.
13. print('\nSERVIDOR INICIADO...IP: ', host, 'PORTA: ', porta,' EM: ',datetime.now().st
     rftime('%d/%m/%Y - %H:%M:%S'))
14.
15. while True:
16.
        conexao, enderecoIP = soquete.accept()
         print('\nSERVIDOR ACESSADO PELO CLIENTE:', enderecoIP, 'EM: ',datetime.now().str
17.
    ftime('%d/%m/%Y - %H:%M:%S'))
18.
        while True:
19.
20.
            mensagem = conexao.recv(2048)
            if not mensagem:
                break
23.
            print('\nIP CLIENTE:', enderecoIP)
24.
             print('MENSAGEN RECEBIDA: ', mensagem.decode(),' - ',datetime.now().strftime
     ('%H:%M:%S'))
25.
26.
         print('CONEXÃO COM O CLIENTE FINALIZADA...', enderecoIP, ' EM: ',datetime.now().
     strftime('%d/%m/%Y - %H:%M:%S'))
27.
         conexao.close()
```

Acrescentamos a linha **3**, importando tudo da classe **datetime**, para registrarmos também a data e a hora das ocorrências.

Na linhas **13, 17, 24 e 26**, foi acrescentado nas mensagens, as funções **datetime.now()** que captura a data do sistema e com o **print** é data e a hora também é impressa, foi utilizada junto a função **strftime()** para formatarmos a data e a hora do sistema, porque a data vem no padrão britânico: ano/mês/dia e os segundos são mostrados em frações de até bilissegundos. Então, utilizamos os parâmetros: **%d** = dia, **%m** = mês, **%Y** = ano com 4 dígitos, **%H** = hora no padrão 24 (se quiser usar padrão 12, utilize **%I)**, **%M** = minutos e **%S** = segundos.

Gravamos o script com o nome: servidor.py.

Vamos agora criar outro script para o cliente:

```
1. # criando um cliente para conexão com o servidor Python
 import socket
 3.
4. host = '127.0.0.1' # mesmo local do servidor
 5. porta = 8800
                      # mesma porta do servidor
 6.
7. soquete = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8. envio = (host,porta)
9. soquete.connect(envio)
10.
11. print('Digite: S e pressione ENTER para encerrar...')
12. print('DIGITE A MENSAGEM: ')
13. mensagem = input()
14.
15. while mensagem not in ('s','S'):
      soquete.send(str(mensagem).encode())
17.
        mensagem = input()
18.
19. soquete.close()
```

Importamos o módulo **socket** e criamos as mesmas variáveis e valores para **host, porta e soquete** (linhas **2, 4, 5 e 7**).

Na linha **8**, foi criada a variável **envio** para enviar o **host** e a **porta** ao servidor.

Na linha 9, o soquete faz a conexão com a função connect.

Nas linhas 11 e 12, são mensagens impressas no cliente.

Na linha **15**, abrimos um loop com **while** sob a condição de enquanto não digitado a letra **s** ou **S** pelo cliente, as mensagens serão enviadas ao **servidor** (linha **16**) através da função **send**.

Quando o **cliente** digitar **s** e pressionar **ENTER**, a conexão será encerrada (linha **19**).

Gravamos o script com o nome: cliente.py.

```
    Traceback (most recent call last):
    File "C:\Users\djsch\OneDrive\Área de Trabalho\cliente.py", line 9, in <module>
    soquete.connect(envio)
```

 ConnectionRefusedError: [WinError 10061] Nenhuma conexão pôde ser feita porque a máq uina de destino as recusou ativamente

Ao executarmos o **script cliente.py** a mensagem acima será exibida porque o servidor não está no ar.

Então, vamos executar primeiro o sript servidor.py:

```
1. SERVIDOR INICIADO...IP: 127.0.0.1 PORTA: 8800 EM: 03/06/2018 - 18:19:03
```

Ao dar um duplo clique no **script servidor.py**, a mensagem acima será exibida no terminal do **servidor**. Registrando o endereço do IP do servidor, a porta e a data e hora em que o servidor subiu.

Agora vamos executar o script cliente.py:

```
1. Digite: S e pressione ENTER para encerrar...
```

2. DIGITE A MENSAGEM:

Ao dar duplo clique no **script cliente.py** um outro terminal será aberto, é como se estivéssemos acessando de outro dispositivo. Esta mensagem acima será exibida no terminal do **cliente.**

Agora vamos ver o que aconteceu no terminal do **servidor**:

```
    SERVIDOR INICIADO...IP: 127.0.0.1 PORTA: 8800 EM: 03/06/2018 - 18:19:03
    SERVIDOR ACESSADO PELO CLIENTE: ('127.0.0.1', 53522) EM: 03/06/2018 - 18:26:36
```

No terminal do **servidor** acima, aparecerá os dados do **cliente** que acessou, o IP, a porta que é um número aleatório gerado pela função AF_INET e a data e hora da conexão.

Agora vamos voltar ao terminal do **cliente** e digitar uma **mensagem**:

```
    Digite: S e pressione ENTER para encerrar...
    DIGITE A MENSAGEM:
    Olá pessoal
    Estou estudando Python
    E enviando uma mensagem ao servidor Python
```

Agora, vamos ver o que ocorreu no terminal do servidor:

```
    SERVIDOR INICIADO...IP: 127.0.0.1 PORTA: 8800 EM: 03/06/2018 - 18:19:03
    SERVIDOR ACESSADO PELO CLIENTE: ('127.0.0.1', 53522) EM: 03/06/2018 - 18:26:36
    IP CLIENTE: ('127.0.0.1', 53522)
    MENSAGEN RECEBIDA: Olá pessoal - 18:37:49
    IP CLIENTE: ('127.0.0.1', 53522)
    MENSAGEN RECEBIDA: Estou estudando Python - 18:38:03
    IP CLIENTE: ('127.0.0.1', 53522)
    MENSAGEN RECEBIDA: E enviando uma mensagem ao servidor Python - 18:38:42
```

OK! Todas mensagens foram recebidas pelo **servidor**.

Agora vamos encerrar o terminal do **cliente**, digitando **s** e pressionando **ENTER**, depois veremos o que ocorreu no terminal do **servidor**:

```
1. SERVIDOR INICIADO...IP: 127.0.0.1 PORTA: 8800 EM: 03/06/2018 - 18:19:03
2.
3. SERVIDOR ACESSADO PELO CLIENTE: ('127.0.0.1', 53522) EM: 03/06/2018 - 18:26:36
4.
5. IP CLIENTE: ('127.0.0.1', 53522)
6. MENSAGEN RECEBIDA: 01á pessoal - 18:37:49
7.
8. IP CLIENTE: ('127.0.0.1', 53522)
9. MENSAGEN RECEBIDA: Estou estudando Python - 18:38:03
10.
11. IP CLIENTE: ('127.0.0.1', 53522)
12. MENSAGEN RECEBIDA: E enviando uma mensagem ao servidor Python - 18:38:42
13. CONEXÃO COM O CLIENTE FINALIZADA... ('127.0.0.1', 53522) EM: 03/06/2018 - 18:48:49
```

A conexão do terminal do **cliente** foi fechada e a ação foi registrada no terminal do **servidor** como visto na linha **13**.

Agora, vamos abrir de novo o terminal do **cliente**, dando um duplo clique no **script cliente.py**, enviando mais uma mensagem e verificar o que aconteceu no terminal do **servidor**:

```
SERVIDOR INICIADO...IP: 127.0.0.1 PORTA: 8800 EM: 03/06/2018 - 18:19:03
1.
 3.
    SERVIDOR ACESSADO PELO CLIENTE: ('127.0.0.1', 53522) EM: 03/06/2018 - 18:26:36
 4.
 5. IP CLIENTE: ('127.0.0.1', 53522)
 6. MENSAGEN RECEBIDA: Olá pessoal - 18:37:49
7.
8. IP CLIENTE: ('127.0.0.1', 53522)
9. MENSAGEN RECEBIDA: Estou estudando Python - 18:38:03
10.
11. IP CLIENTE: ('127.0.0.1', 53522)
12. MENSAGEN RECEBIDA: E enviando uma mensagem ao servidor Python - 18:38:42
13. CONEXÃO COM O CLIENTE FINALIZADA... ('127.0.0.1', 53522) EM: 03/06/2018 - 18:48:49
14.
15. SERVIDOR ACESSADO PELO CLIENTE: ('127.0.0.1', 53616) EM: 03/06/2018 - 18:56:35
16.
17.
    IP CLIENTE: ('127.0.0.1', 53616)
18. MENSAGEN RECEBIDA: Oi, pessoal estou de volta! - 18:57:01
```

Novamente, o **servidor** registrou a nova conexão do **cliente** (linha **15**) e a mensagem foi recebida! (linha **18**).

SAIBA MAIS...

Dê uma olhada nos links abaixo para saber mais sobre a linguagem Python:

https://www.python.org/doc/ (https://www.python.org/doc/)

https://wiki.python.org/moin/PythonBooks (https://wiki.python.org/moin/PythonBooks)

Neste tópico criamos um exemplo de aplicação em Python, trocando mensagens entre uma conexão cliente e servidor.

Ouiz

Exercício Final

Recebendo uma conexão em um servidor Python

INICIAR >

Referências

SUMMERFIELD, M. *Programação em Python 3*: Uma introdução completa à linguagem Python. Rio de Janeiro Alta Books, 2012. 495 p.

MENEZES, N. N. C. *Introdução à programação com Python:* algoritmos e lógica de programação para iniciantes. 2. ed. São Paulo: Novatec, 2014. 328 p.

SWEIGART, AL. *Automatize tarefas maçantes com Python:* programação prática para verdadeiros iniciantes. São Paulo: Novatec, 2015. 568 p.

PYTHON, doc. Disponível em: https://www.python.org/doc/. Acesso em: Junho/2018.

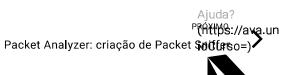
PYTHON, books. Disponível em: https://wiki.python.org/moin/PythonBooks. Acesso em: Junho/2018.



Avalie este tópico







® Todos os direitos reservados