


1. Este apartado crea el socket con el puerto 12000, en mi caso uso localhost para poder usar la conexión cliente con la misma máquina.

2. Para añadir la conexión con servidor deberemos añadir "socketCliente" y como estamos usando el modo UDP necesitaremos usar "(AF_INET, SOCK_DGRAM)"

3. Por último usamos "mensaje" para añadir un simple mensaje que podrá ver el cliente una vez inicie la programación



AF_INET = IPv4
SOCK_DGRAM = UDP



```
# /usr/bin/env python
# -*- coding: utf-8 -*-
from socket import *

serviNombre = 'localhost'
serviPuerto = 12000
socketCliente = socket(AF_INET, SOCK_DGRAM)
mensaje = input('Mete una línea en minúsculas: ')
```


```
socketCliente.sendto(mensaje.encode(), (serviNombre, serviPuerto))
mensajeMayu, serviDir = socketCliente.recvfrom(2048)
print (mensajeMayu.decode())
socketCliente.close()
```



1. Una vez el cliente haya recibido el mensaje deberemos usar socketCliente ya que este código envía un mensaje al servidor identificado por serviNombre y serviPuerto

1. Este código recibe un mensaje de hasta 2048 bytes del servidor, decodifica el mensaje recibido, lo imprime en la consola y finalmente cierra el socket

1. Este código crea un servidor UDP que escucha en el puerto 12000. El servidor está listo para recibir mensajes en minúsculas que probablemente va a convertir en mayúsculas más adelante. Utiliza el protocolo IPv4 (AF_INET) y UDP (SOCK_DGRAM)





```
from socket import *

serviPuerto = 12000
socketServidor = socket(AF_INET, SOCK_DGRAM)
socketServidor.bind(('', serviPuerto))
print('Soy Mayusculator y espero cadenas en minúscula')

while True:
    mensaje, clienteDir = socketServidor.recvfrom(2048)
    mensajeMay = mensaje.upper()
    socketServidor.sendto(mensajeMay, clienteDir)
```

1. Este código mantiene al servidor escuchando continuamente por mensajes entrantes. Cada vez que un cliente envía un mensaje, el servidor lo recibe y también obtiene la dirección del cliente que envió el mensaje

- 
- 
1. El mensaje recibido desde el cliente, lo convierte a mayúsculas y luego lo envía de vuelta al cliente
 2. El servidor usa el método sendto() para enviar el mensaje de vuelta, asegurándose de que el mensaje se envíe a la dirección del cliente que lo solicitó
 3. Si el mensaje está en formato de texto, debe codificarse en bytes antes de enviarlo a través de la red.

CONEXION ENTRE SERVIDOR Y CLIENTE EN UDP

1º PASO: Iniciar servidor

Para ello, ejecutaremos el comando de Servidor UDP

```
/bin/python3.12 /home/usuario/PYTHON/ServerUDP.py
○ usuario@usuario-Legion-5-15IAH7H:~/VS$ /bin/python3.12 /home/usuario/PYTHON/ServerUDP.py
Soy Mayusculador y espero cadenas en minúscula
```

2º PASO: Iniciar cliente

Para ello, ejecutaremos el archivo de cliente desde la terminal

```
○ usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ python3 clienteUDP.py
Mete una linea en minusculas: █
```

3º PASO: Comprobacion

Usaremos “hola” como comprobacion e iniciaremos

```
● usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ python3 clienteUDP.py
Mete una linea en minusculas: hola
HOLA
○ usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ █
```

Programas cliente y servidor UDP interactuando

Servidor

Crea socket, **puerto= x**
socketServidor =
socket(AF_INET, SOCK_DGRAM)

lee el segmento UDP de
socketServidor

Escribe respuesta en
socketServidor
indicando la dirección
del cliente y el
número de puerto

Cliente

Crea socket
socketCliente =
socket(AF_INET, SOCK_DGRAM)

Crea datagrama con la IP del servidor
y port=x. Envía datagrama vía
socketCliente

lee datagrama de
socketCliente

cierra
socketCliente

Aqui tenemos un ejemplo resumido de como seria una conexion UDP entre estos dos componentes

1. Este código configura un servidor TCP que escucha en el puerto 12000. Usa IPv4 y el protocolo TCP para recibir conexiones de clientes

2. Una vez configurado, el servidor entra en modo de espera y está listo para aceptar conexiones entrantes de clientes. Una vez que un cliente se conecta, el servidor procesará las solicitudes.

1. Este código mantiene al servidor en estado de espera continuamente para aceptar nuevas conexiones de clientes. Cada vez que un cliente se conecta, se crea un nuevo socket dedicado a esa conexión, y el servidor puede comunicarse con ese cliente

```
#!/usr/bin/env python
#-*- coding: utf-8 -*-

from socket import *

serviPuerto = 12000
socketServidor = socket(AF_INET, SOCK_STREAM)
socketServidor.bind(('', serviPuerto))
socketServidor.listen(1)
print ('Soy Mayusculator y espero cadenas en minúscula')

while True:
    socketConex, clienteDir = socketServidor.accept()
    mensaje = socketConex.recv(1024)
    mensajeMay = mensaje.upper()
    socketConex.send(mensajeMay)
    socketConex.close()
```

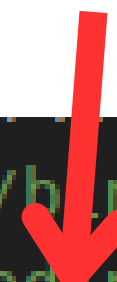
Aquí podemos ver la configuración del mensaje y lo que recibirá dicho cliente.

Ejemplo: Un cliente envía el mensaje "hola", el servidor lo recibe, lo convierte a "HOLA", y se lo envía de vuelta al cliente, cerrando luego la conexión

1. Este código crea un cliente TCP que se conecta a un servidor en localhost en el puerto 12000.

2. Una vez conectados, el cliente y el servidor pueden intercambiar datos utilizando el socket que se crea "socket(AF_INET, SOCK_STREAM)"

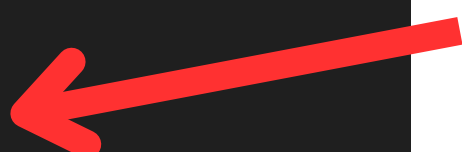
SOCK_STREAM = TCP



```
#!/usr/bin/env python
#-*- coding: utf-8 -*-
from socket import *

serviNombre = 'localhost'
serviPuerto = 12000
socketCliente = socket(AF_INET, SOCK_STREAM)
socketCliente.connect((serviNombre, serviPuerto))

mensaje = input('Mete una linea en minusculas: ')
socketCliente.send(mensaje.encode())
mensajeMayu = socketCliente.recv(1024)
print (mensajeMayu.decode())
socketCliente.close()
```

1. Solicita al usuario que ingrese una línea de texto en minúsculas
 2. Envía esa línea al servidor después de convertirla a bytes
 3. Recibiremos una respuesta del servidor, que se espera que esté en mayúsculas
 4. Imprime la respuesta en la consola y cierra la conexión
- 

CONEXION ENTRE SERVIDOR Y CLIENTE EN TCP

1º PASO: Iniciar servidor

Para ello, ejecutaremos el comando de Servidor TCP

```
/bin/python3.12 /home/usuario/PYTHON/ServerTCP.py  
● usuario@usuario-Legion-5-15IAH7H:~/VS$ /bin/python3.12 /home/usuario/PYTHON/ServerTCP.py  
Soy Mayusculator y espero cadenas en minúscula
```

2º PASO: Iniciar cliente

Para ello, ejecutaremos el archivo de cliente desde la terminal

```
● usuario@usuario-Legion-5-15IAH7H:~/VS$ cd ../PYTHON/  
○ usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ python3 clienteTCP.py  
Mete una linea en minusculas: █
```

3º PASO: Comprobacion

Usaremos “hola” como comprobacion e iniciaremos

```
● usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ python3 clienteTCP.py  
Mete una linea en minusculas: hola  
HOLA  
○ usuario@usuario-Legion-5-15IAH7H:~/PYTHON$ █
```

Programas cliente y servidor TCP interactuando

Servidor

Cliente

crea socket `puerto=x`,
para solicitudes entrantes:

`socketServidor =
socket()`

Espera solicitud
de conexión

`socketConex =
socketServidor.accept()`

lee solicitud de
`socketConex`

escribe respuesta a
`socketConex`

cierra
`socketConex`

Establecimiento
de conexión TCP

crea socket conectando a
`Servidor, port=x`
`socketCliente =
socket()`

envía solicitud usando
`socketCliente`

lee respuesta desde
`socketCliente`

cierra
`socketCliente`

Aqui tenemos un ejemplo resumido de como seria una conexion TCP entre estos dos componentes