UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

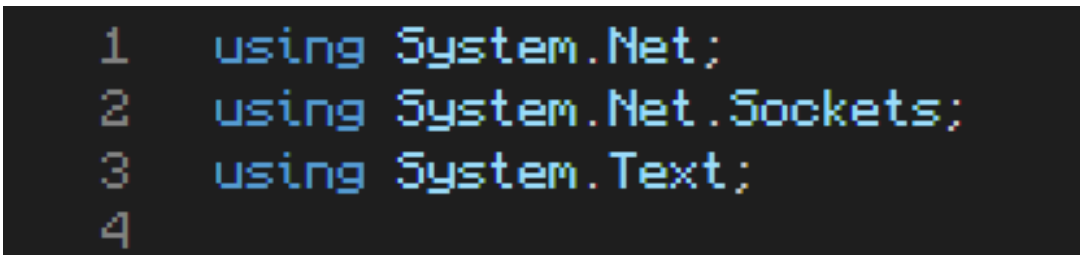
FACULTAD DE INFORMÁTICA

Flujo de Aplicación P2P

Jorge Francisco Acosta Alcalá

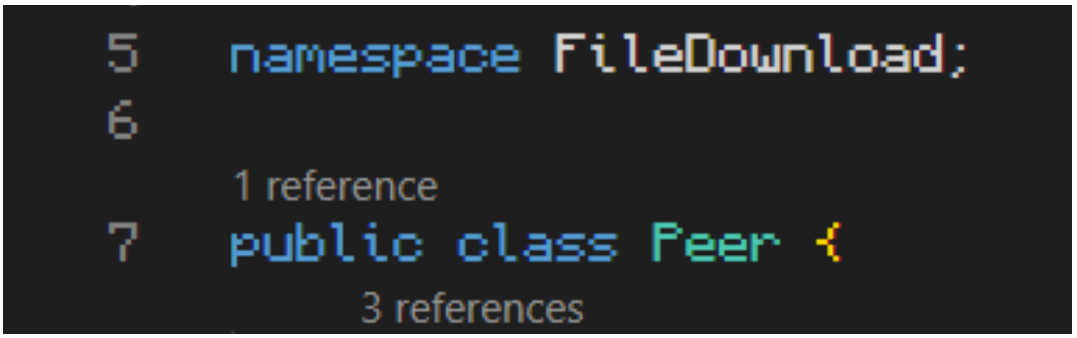
SOFT 18: G36

19/08/2024



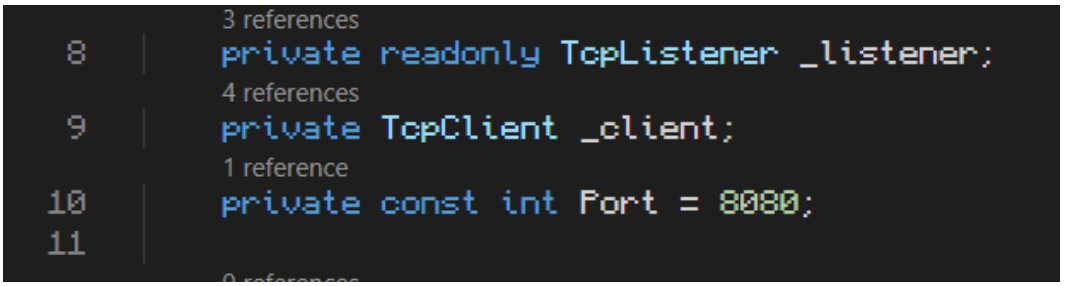
```
1   using System.Net;
2   using System.Net.Sockets;
3   using System.Text;
4
```

Se importan las bibliotecas que se usarán en el código.



```
5   namespace FileDownload;
6
        1 reference
7   public class Peer {
            3 references
```

Se define el namespace para organizar el código, y también se define la clase peer que es donde irá todo el código

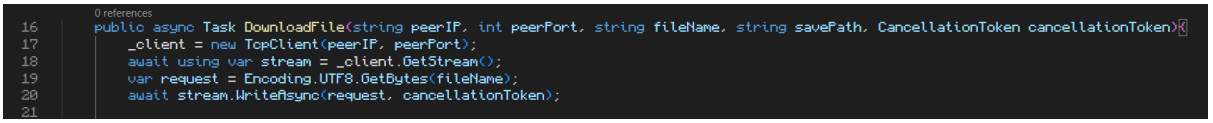


```
        3 references
8       private readonly TcpListener _listener;
        4 references
9       private TcpClient _client;
        1 reference
10      private const int Port = 8080;
11
```

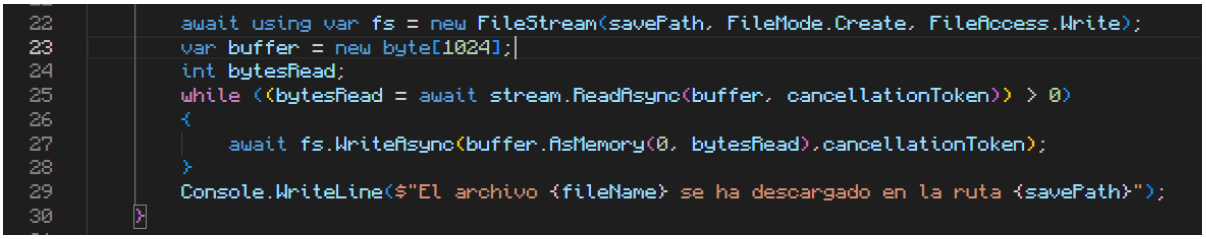
Primero se declara el Listener que es para escuchar conexiones entrantes, luego se crea el Client que es para manejar la conexión con el cliente. al final se define un puerto que el código estará escuchando constantemente o al que se conectara.

```
        0 references
12      public Peer(){
13          _listener = new TcpListener(IPAddress.Any, Port);
14      }
15
```

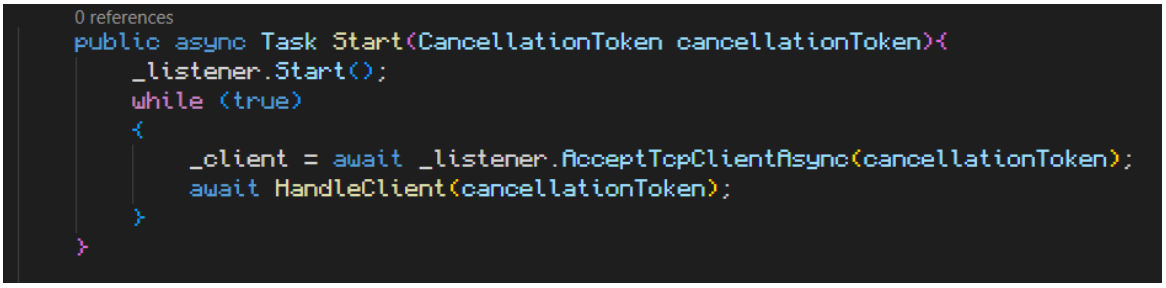
Es el constructor, hace que se active el Listener para que escuche cualquier dirección IP en el puerto que le asignamos antes.



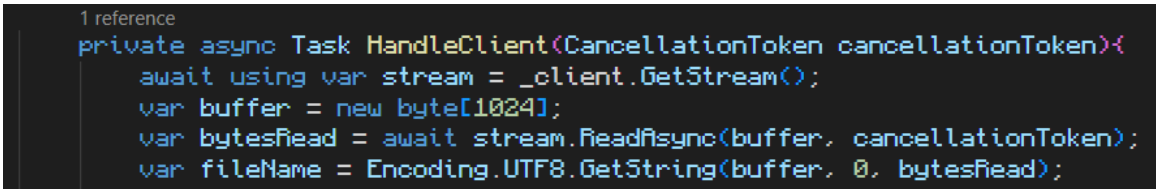
Este es el método para descargar los archivos, primero conecta al cliente en el puerto y con la ip que le especificamos, luego con el código stream conseguimos los datos de la conexión mediante TCP, luego convertimos en una secuencia de bytes el nombre del archivo que queremos descargar, al final se le envía el nombre del archivo al peer.



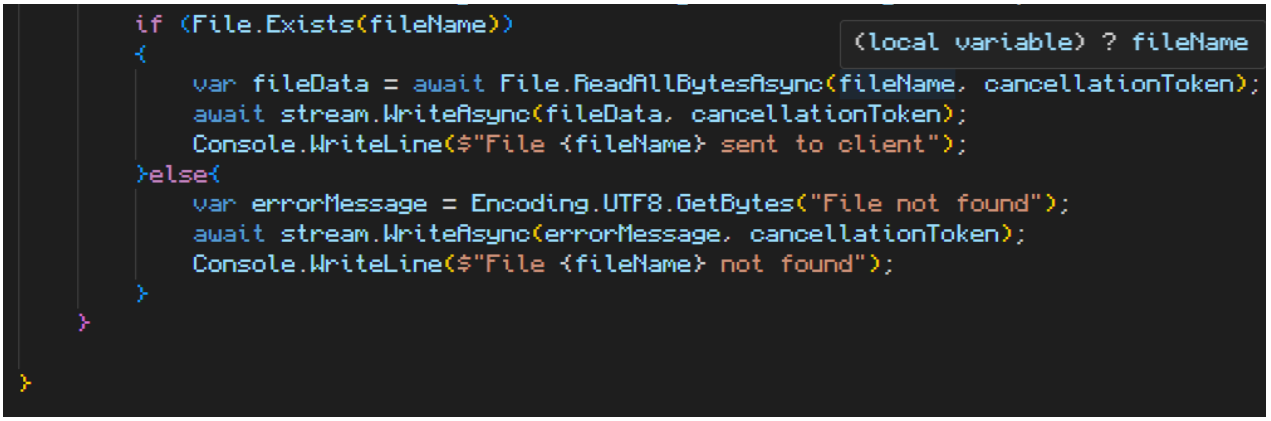
Luego le damos permisos para crear y escribir en archivos y creamos el archivo en la ruta especificada. Creamos un buffer para leer los datos y otra variable int para guardar los bytes que nos enviaran. Luego en el while se leen los datos hasta que ya no haya más datos y mientras los va escribiendo en el archivo que creamos. al finalizar de leerlos y escribirlos se envía un mensaje a consola indicando el nombre del archivo y donde se guardo.



El método start es para la conexión del servidor, primero se inicia el Listener para escuchar conexiones, se entra en un bucle infinito hasta que encuentre una conexión, y cuando la encuentra le asigna el Client y lo envía al método HandleClient.



En el método HandleClient se manejan las solicitudes del cliente, primero recibimos el flujo de datos con el método GetStream, creamos un buffer y un bytesread para recibir la información, luego la decodificamos para conseguir el nombre del archivo que se enviará al cliente.



Luego de recibir el nombre del archivo que enviaremos, revisamos si existe, si existe, leemos los bytes del archivo y los guardamos en una variable, para luego enviarlos en el flujo de datos con el método stream. Al final se escribe en consola que archivo fue enviado al cliente.

Ahora, si el archivo no existe primero codificamos un mensaje de error que se le enviará al cliente mediante el método stream, y se escribe el nombre del archivo que no fue encontrado en consola.