

Getting Error in Login() method in FtpConnection Class

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2006-02/msg00067.html>

- *From:* ruju00@xxxxxxxxxx
 - *Date:* 1 Feb 2006 02:02:30 -0800
-

I am getting an error in Login() method of the following class
FtpConnection

```
public class FtpConnection
{
    public class FtpException : Exception
    {
        public FtpException(string message) : base(message){}
        public FtpException(string message, Exception innerException) :
base(message, innerException){}
    }

    private static int BUFFER_SIZE = 512;
    private static Encoding ASCII = Encoding.ASCII;

    private bool verboseDebugging = false;

    private string server = "localhost";
    //private string server = "81.19.59.132";
    private string remotePath = ".";
    private string username = "anonymous";
    private string password = "anonymous@xxxxxxxxxxxxxxxxxx";
    private string message = null;
    private string result = null;

    private int port = 21;
    private int bytes = 0;
    private int resultCode = 0;

    private bool loggedin = false;
    private bool binMode = false;

    private Byte[] buffer = new Byte[BUFFER_SIZE];
    private Socket clientSocket = null;

    private int timeoutSeconds = 10;

    public FtpConnection()
    {
    }

    public FtpConnection(string server, string username, string password)
    {
        this.server = server;
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
        this.username = username;
        this.password = password;
    }

    public FtpConnection(string server, string username, string password,
int timeoutSeconds, int port)
    {
        this.server = server;
        this.username = username;
        this.password = password;
        this.timeoutSeconds = timeoutSeconds;
        this.port = port;
    }

    //                                Display all communications to the debug log
    public bool VerboseDebugging
    {
        get
        {
            return this.verboseDebugging;
        }
        set
        {
            this.verboseDebugging = value;
        }
    }

    //                                Remote server port. Typically TCP 21

    public int Port
    {
        get
        {
            return this.port;
        }
        set
        {
            this.port = value;
        }
    }

    //                                Timeout waiting for a response from server, in seconds.

    public int Timeout
    {
        get
        {
            return this.timeoutSeconds;
        }
        set
        {
            this.timeoutSeconds = value;
        }
    }

    //                                Gets and Sets the name of the FTP server.

    public string Server
    {
        get
        {
            return this.server;
        }
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
    }
    set
    {
        this.server = value;
    }
}

//          Gets and Sets the port number.

public int RemotePort
{
    get
    {
        return this.port;
    }
    set
    {
        this.port = value;
    }
}

//          GetS and Sets the remote directory.

public string RemotePath
{
    get
    {
        return this.remotePath;
    }
    set
    {
        this.remotePath = value;
    }
}

//          Gets and Sets the username.

public string Username
{
    get
    {
        return this.username;
    }
    set
    {
        this.username = value;
    }
}

//          Gets and Set the password.

public string Password
{
    get
    {
        return this.password;
    }
    set
    {
        this.password = value;
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
    }

    // If the value of mode is true, set binary mode for download
else, Ascii mode.

    public bool BinaryMode
    {
        get
        {
            return this.binMode;
        }
        set
        {
            if ( this.binMode == value ) return;

            if ( value )
                sendCommand("TYPE I");

            else
                sendCommand("TYPE A");

            if ( this.resultCode != 200 ) throw new
FtpException(result.Substring(4));
        }
    }

    // Login to the remote server.

    public void Login()
    {
        if (this.loggedin) this.Close();

        Debug.WriteLine("Opening connection to " + this.server, "FtpClient"
);

        IPAddress addr = null;
        IPEndPoint ep = null;

        try
        {
            this.clientSocket = new Socket( AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp );
            addr = Dns.Resolve(this.server).AddressList[0];
            ep = new IPEndPoint( addr, this.port );
            this.clientSocket.Connect(ep);
        }
        catch(Exception ex)
        {
            if ( this.clientSocket != null && this.clientSocket.Connected )
this.clientSocket.Close();

            throw new FtpException("Couldn't connect to remote server",ex);
        }

        this.readResponse();

        if(this.resultCode != 220)
        {
            this.Close();
            throw new FtpException(this.result.Substring(4));
        }
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
        this.sendCommand( "USER " + username );

        if( !(this.resultCode == 331 || this.resultCode == 230) )
        {
            this.cleanup();
            throw new FtpException(this.result.Substring(4));
        }

        if( this.resultCode != 230 )
        {
            this.sendCommand( "PASS " + password );

            if( !(this.resultCode == 230 || this.resultCode == 202) )
            {
                this.cleanup();
                throw new FtpException(this.result.Substring(4));
            }
        }

        this.loggedin = true;

        Debug.WriteLine( "Connected to " + this.server, "FtpClient" );

        this.ChangeDir(this.remotePath);
    }

    //          Close the FTP connection.

    public void Close()
    {
        Debug.WriteLine("Closing connection to " + this.server, "FtpClient"

);

        if( this.clientSocket != null )
        {
            this.sendCommand("QUIT");
        }

        this.cleanup();
    }

    //          Return a string array containing the remote directory's f
list.

    public string[] GetFileList()
    {
        return this.GetFileList("*.");
    }

    //          Return a string array containing the remote directory's f
list.

    public string[] GetFileList(string mask)
    {
        if ( !this.loggedin ) this.Login();

        Socket cSocket = createDataSocket();

        this.sendCommand("NLST " + mask);

        if(!(this.resultCode == 150 || this.resultCode == 125)) throw new
FtpException(this.result.Substring(4));
```

Getting Error in Login() method in FtpConnection Class

```
this.message = "";

DateTime timeout = DateTime.Now.AddSeconds(this.timeoutSeconds);

while( timeout > DateTime.Now )
{
    int bytes = cSocket.Receive(buffer, buffer.Length, 0);
    this.message += ASCII.GetString(buffer, 0, bytes);

    if ( bytes < this.buffer.Length ) break;
}

string[] msg = this.message.Replace("\r","").Split('\n');

cSocket.Close();

if ( this.message.IndexOf( "No such file or directory" ) != -1 )
    msg = new string[]{};

this.readResponse();

if ( this.resultCode != 226 )
    msg = new string[]{};
//    throw new FtpException(result.Substring(4));
return msg;
}

//                                Return the size of a file.

public long GetFileSize(string fileName)
{
    if ( !this.loggedin ) this.Login();

    this.sendCommand("SIZE " + fileName);
    long size=0;

    if ( this.resultCode == 213 )
        size = long.Parse(this.result.Substring(4));

    else
        throw new FtpException(this.result.Substring(4));

    return size;
}

//                                Download a file to the Assembly's local directory,
//                                keeping the same file name.

public void Download(string remFileName)
{
    this.Download(remFileName,"",false);
}

//                                Download a remote file to the Assembly's local directory,
//                                keeping the same file name, and set the resume flag.

public void Download(string remFileName,Boolean resume)
{
    this.Download(remFileName,"",resume);
}
```

Getting Error in Login() method in FtpConnection Class

```
//          Download a remote file to a local file name which can include
//          a path. The local file name will be created or overwritten
//          but the path must exist.

public void Download(string remFileName,string locFileName)
{
    this.Download(remFileName,locFileName,false);
}

//          Download a remote file to a local file name which can include
//          a path, and set the resume flag. The local file name will be
//          created or overwritten, but the path must exist.

public void Download(string remFileName,string locFileName,Boolean
resume)
{
    if ( !this.loggedin ) this.Login();

    this.BinaryMode = true;

    Debug.WriteLine("Downloading file " + remFileName + " from " +
server + "/" + remotePath, "FtpClient" );

    if (locFileName.Equals(""))
    {
        locFileName = remFileName;
    }

    FileStream output = null;

    if ( !File.Exists(locFileName) )
        output = File.Create(locFileName);

    else
        output = new FileStream(locFileName,FileMode.Open);

    Socket cSocket = createDataSocket();

    long offset = 0;

    if ( resume )
    {
        offset = output.Length;

        if ( offset > 0 )
        {
            this.sendCommand( "REST " + offset );
            if ( this.resultCode != 350 )
            {
                //Server doesnt support resuming
                offset = 0;
                Debug.WriteLine("Resuming not supported:" + resultCode,
"FtpClient" );
            }
            else
            {
                Debug.WriteLine("Resuming at offset " + offset, "FtpClient");
                output.Seek( offset, SeekOrigin.Begin );
            }
        }
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
this.sendCommand("RETR " + remFileName);

if ( this.resultCode != 150 && this.resultCode != 125 )
{
    throw new FtpException(this.result.Substring(4));
}

DateTime timeout = DateTime.Now.AddSeconds(this.timeoutSeconds);

while ( timeout > DateTime.Now )
{
    this.bytes = cSocket.Receive(buffer, buffer.Length, 0);
    output.Write(this.buffer,0,this.bytes);

    if ( this.bytes <= 0 )
    {
        break;
    }
}

output.Close();

if ( cSocket.Connected ) cSocket.Close();

this.readResponse();

if( this.resultCode != 226 && this.resultCode != 250 )
    throw new FtpException(this.result.Substring(4));
}

//          Upload a file.

public void Upload(string fileName)
{
    this.Upload(fileName,false);
}

//          Upload a file and set the resume flag.

public void Upload(string fileName, bool resume)
{
    if ( !this.loggedin ) this.Login();

    Socket cSocket = null ;
    long offset = 0;

    if ( resume )
    {
        try
        {
            this.BinaryMode = true;

            offset = GetFileSize( Path.GetFileName(fileName) );
        }
        catch(Exception)
        {
            // file not exist
            offset = 0;
        }
    }

    // open stream to read file
```

Getting Error in Login() method in FtpConnection Class

```
FileStream input = new FileStream(fileName, FileMode.Open);

if ( resume && input.Length < offset )
{
    // different file size
    Debug.WriteLine("Overwriting " + fileName, "FtpClient");
    offset = 0;
}
else if ( resume && input.Length == offset )
{
    // file done
    input.Close();
    Debug.WriteLine("Skipping completed " + fileName + " - turn resum
off to not detect.", "FtpClient");
    return;
}

// dont create untill we know that we need it
cSocket = this.createDataSocket();

if ( offset > 0 )
{
    this.sendCommand( "REST " + offset );
    if ( this.resultCode != 350 )
    {
        Debug.WriteLine("Resuming not supported", "FtpClient");
        offset = 0;
    }
}

this.sendCommand( "STOR " + Path.GetFileName(fileName) );

if ( this.resultCode != 125 && this.resultCode != 150 ) throw new
FtpException(result.Substring(4));

if ( offset != 0 )
{
    Debug.WriteLine("Resuming at offset " + offset, "FtpClient" );

    input.Seek(offset, SeekOrigin.Begin);
}

Debug.WriteLine( "Uploading file " + fileName + " to " + remotePath,
"FtpClient" );

while ((bytes = input.Read(buffer,0,buffer.Length)) > 0)
{
    cSocket.Send(buffer, bytes, 0);
}

input.Close();

if (cSocket.Connected)
{
    cSocket.Close();
}

this.readResponse();

if( this.resultCode != 226 && this.resultCode != 250 ) throw new
FtpException(this.result.Substring(4));
}
```

Getting Error in Login() method in FtpConnection Class

```
// Upload a directory and its file contents
public void UploadDirectory(string path, bool recurse)
{
    this.UploadDirectory(path, recurse, "*.*");
}

// Upload a directory and its file contents
public void UploadDirectory(string path, bool recurse, string mask)
{
    string[] dirs = path.Replace("/", @"\").Split('\\');
    string rootDir = dirs[ dirs.Length - 1 ];

    // make the root dir if it does not exist
    if ( this.GetFilesList(rootDir).Length < 1 ) this.MakeDir(rootDir);

    this.ChangeDir(rootDir);

    foreach ( string file in Directory.GetFiles(path, mask) )
    {
        this.Upload(file, true);
    }
    if ( recurse )
    {
        foreach ( string directory in Directory.GetDirectories(path) )
        {
            this.UploadDirectory(directory, recurse, mask);
        }
    }

    this.ChangeDir("..");
}

// Delete a file from the remote FTP server.
public void DeleteFile(string fileName)
{
    if ( !this.loggedin ) this.Login();

    this.sendCommand( "DELE " + fileName );

    if ( this.resultCode != 250 ) throw new
FtpException(this.result.Substring(4));

    Debug.WriteLine( "Deleted file " + fileName, "FtpClient" );
}

// Rename a file on the remote FTP server.
public void RenameFile(string oldFileName, string newFileName, bool
overwrite)
{
    if ( !this.loggedin ) this.Login();

    this.sendCommand( "RNFR " + oldFileName );

    if ( this.resultCode != 350 ) throw new
FtpException(this.result.Substring(4));

    if ( !overwrite && this.GetFilesList(newFileName).Length > 0 ) throw
```

Getting Error in Login() method in FtpConnection Class

```
new FtpException("File already exists");

        this.sendCommand( "RNTO " + newFileName );

        if ( this.resultCode != 250 ) throw new
FtpException(this.result.Substring(4));

        Debug.WriteLine( "Renamed file " + oldFileName + " to " +
newFileName, "FtpClient" );
    }

    //          Create a directory on the remote FTP server.

    public void MakeDir(string dirName)
    {
        if ( !this.loggedin ) this.Login();

        this.sendCommand( "MKD " + dirName );

        if ( this.resultCode != 250 && this.resultCode != 257 ) throw new
FtpException(this.result.Substring(4));

        Debug.WriteLine( "Created directory " + dirName, "FtpClient" );
    }

    //          Delete a directory on the remote FTP server.

    public void RemoveDir(string dirName)
    {
        if ( !this.loggedin ) this.Login();

        this.sendCommand( "RMD " + dirName );

        if ( this.resultCode != 250 ) throw new
FtpException(this.result.Substring(4));

        Debug.WriteLine( "Removed directory " + dirName, "FtpClient" );
    }

    //          Change the current working directory on the remote FTP se

    public void ChangeDir(string dirName)
    {
        if( dirName == null || dirName.Equals(".") || dirName.Length == 0 )
        {
            return;
        }

        if ( !this.loggedin ) this.Login();

        this.sendCommand( "CWD " + dirName );

        if ( this.resultCode != 250 ) throw new
FtpException(result.Substring(4));

        this.sendCommand( "PWD" );

        if ( this.resultCode != 257 ) throw new
FtpException(result.Substring(4));

        // gonna have to do better than this....
        this.remotePath = this.message.Split(' ')[1];
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
        Debug.WriteLine( "Current directory is " + this.remotePath,
"FtpClient" );
    }

    private void readResponse()
    {
        this.message = "";
        this.result = this.readLine();

        if ( this.result.Length > 3 )
            this.resultCode = int.Parse( this.result.Substring(0,3) );
        else
            this.result = null;
    }

    private string readLine()
    {
        while(true)
        {
            this.bytes = clientSocket.Receive( this.buffer, this.buffer.Length
0 );

            this.message += ASCII.GetString( this.buffer, 0, this.bytes );

            if ( this.bytes < this.buffer.Length )
            {
                break;
            }
        }

        string[] msg = this.message.Split('\n');

        if ( this.message.Length > 2 )
            this.message = msg[ msg.Length - 2 ];

        else
            this.message = msg[0];

        if ( this.message.Length > 4 &&
!this.message.Substring(3,1).Equals(" ") ) return this.readLine();

        if ( this.verboseDebugging )
        {
            for(int i = 0; i < msg.Length - 1; i++)
            {
                Debug.Write( msg[i], "FtpClient" );
            }
        }

        return message;
    }

    private void sendCommand(String command)
    {
        if ( this.verboseDebugging ) Debug.WriteLine(command,"FtpClient");

        Byte[] cmdBytes = Encoding.ASCII.GetBytes( ( command + "\r\n"
).ToCharArray() );

        clientSocket.Send( cmdBytes, cmdBytes.Length, 0);
        this.readResponse();
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
it. // when doing data transfers, we need to open another socket

private Socket createDataSocket()
{
    this.sendCommand("PASV");

    if ( this.resultCode != 227 ) throw new
FtpException(this.result.Substring(4));

    int index1 = this.result.IndexOf('(');
    int index2 = this.result.IndexOf(')');

    string ipData = this.result.Substring(index1+1,index2-index1-1);

    int[] parts = new int[6];

    int len = ipData.Length;
    int partCount = 0;
    string buf="";

    for (int i = 0; i < len && partCount <= 6; i++)
    {
        char ch = char.Parse( ipData.Substring(i,1) );

        if ( char.IsDigit(ch) )
            buf+=ch;

        else if (ch != ',')
            throw new FtpException("Malformed PASV result: " + result
+ this.result, ex);

        if ( ch == ',' || i+1 == len )
        {
            try
            {
                parts[partCount++] = int.Parse(buf);
                buf = "";
            }
            catch (Exception ex)
            {
                throw new FtpException("Malformed PASV result (no
parts[3];

                int port = (parts[4] << 8) + parts[5];

                Socket socket = null;
                IPEndPoint ep = null;

                try
                {
                    socket = new
Socket(AddressFamily.InterNetwork,SocketType.Stream,ProtocolType.Tcp);
                    ep = new IPEndPoint(Dns.Resolve(ipAddress).AddressList[0], port);
                    socket.Connect(ep);
                }
            }
        }
    }

    string ipAddress = parts[0] + "."+ parts[1]+ "." + parts[2] + "." +
parts[3];

    int port = (parts[4] << 8) + parts[5];

    Socket socket = null;
    IPEndPoint ep = null;

    try
    {
        socket = new
Socket(AddressFamily.InterNetwork,SocketType.Stream,ProtocolType.Tcp);
        ep = new IPEndPoint(Dns.Resolve(ipAddress).AddressList[0], port);
        socket.Connect(ep);
    }
}
```

Getting Error in Login() method in FtpConnection Class

```
        catch(Exception ex)
        {
            // doubtful...
            if ( socket != null && socket.Connected ) socket.Close();

            throw new FtpException("Can't connect to remote server", ex);
        }

        return socket;
    }

    // Always release those sockets.
    private void cleanup()
    {
        if ( this.clientSocket!=null )
        {
            this.clientSocket.Close();
            this.clientSocket = null;
        }
        this.loggedin = false;
    }

    // Destuctor

    ~FtpConnection()
    {
        this.cleanup();
    }
}
```

Here I am calling this class through

```
FtpConnection ftp = new
FtpConnection("servername", "username", "password");
```

```
ftp.Login();
ftp.Upload(@filename);
```

But it gives the error `Error System.NullReferenceException: Object reference not set to an instance of an object.`

Please help me here.....

.