آموزش کار با کنترل WinSock

کنترل WinSock به شما اجازه می دهد که به یک ماشین راه دور متصل شده و داده ها را با استفاده از پروتکل های UDP و TCP مبادله کنید. هر دو پروتکل می توانند برای برنامه های کلاینت سرور استفاده شوند. مانند کنترل Timer این کنترل هم در زمان اجرا دیده نمی شود.

موارد استفاده

- ساختن یک برنامه کلاینت که اطلاعات کاربر را قبل از ارسال به سرور جمع آوری می کند.
- ساختن یک برنامه سرور که به عنوان نقطه مرکزی یک مجموعه برای داده های چندین کاربر می باشد.
- ساختن یک برنامه Chat

انتخاب یک پروتکل

وقتی از کنترل WinSock استفاده می کنید، باید به این مسئله رسیدگی کنید که کدامیک از پروتکلهای UDP و TCP را می خواهید استفاده کنید. تفاوت اصلی بین این دو در حالت اتصال آنها می باشد:

- پروتکل TCP یک پروتکل اتصال-پایه است و مانند تلفن کاربر باید قبل از اقدام به ارسال اطلاعات یک ارتباط ایجاد کند.
- پروتکل UDP یک پروتکل بدون اتصال است و مبادله اطلاعات بین دو کامپیوتر مانند ارسال پیغام است: یک پیغام از یک کامپیوتر برای یکی دیگر ارسال می شود اما ارتباط بین آن دو واضح نیست. حداکثر اندازه داده ارسال شده بصورت منفرد، بوسیله شبکه تعیین می شود.

طبیعت برنامه ای که خواهید ساخت، عموماً تعیین می کند که چه پروتکلی را انتخاب کنید. در اینجا تعدادی سوال مطرح شده است که می تواند به شما در انتخاب پروتکل برای برنامه کمک کند:

۱- آیا برنامه شما وقتی که پیغامی را دریافت و یا ارسال می کند به پیغام تصدیق از طرف سرور و یاکلاینت احتیاج خواهد داشت؟ اگر داشت برای برقراری یک اتصال صریح به پروتکل TCP احتیاج دارید.

۲- داده های شما خیلی بزرگ هستند (مانند یک فایل تصویری یا صوتی)؟ هنگامی که یک اتصال ساخته شد، پروتکل TCP اتصال را نگه می دارد و از درستی داده ها مطمئن می شود. این اتصال، هرچند که کاربران منابع بیشتری را پردازش می کنند، آن را هزینه بر می سازد.

۳- داده ها بطور متناوب ارسال می شوند یا در یک نشست؟ برای مثال اگر شما برنامه ای ساخته اید که پس از اتمام وظایف به کامپیوترها هشدار می دهد، پروتکل UDP به نظر شایسته تر خواهد بود. پروتکل UDP برای ارسال مقادیر کمی از داده استفاده بیشتری دارد.

تنظیم پروتکل

برای تنظیم پروتکلی که برنامه شما از آن استفاده خواهد کرد: در حالت طراحی، در پنجره خواص، روی Protocol کلیک کنید و یکی از دو حالت sckTCPProtocol یا sckUDPProtocol را انتخاب کنید. شما اغلب می توانید خاصیت Protocol را در کد تنظیم کنید، مانند مثال زیر:

```
Winsock1.Protocol = sckTCPProtocol
```

مشخص کردن نام کامپیوتر شما

برای اتصال به یک کامپیوتر دور شما باید بدانید IP و یا نام دوستانه آن را بدانید. آدرس IP یک سری از عددهای سه رقمی است که بوسیله نقطه از یکدیگر جدا شده اند. به خاطر سپردن نام دوستانه یک کامپیوتر بسیار ساده تر از IP آن خواهد بود.

برای پیدا کردن نام دوستانه کامپیوترتان:

۱- دکمه Start را کلیک کنید

۲- در گزینه Setting روی Control Panel کلیک کنید

۳- روی آیکون Network کلیک کنید

۴- برگه Identification را کلیک کنید

۵- نام کامپیوترتان را در جعبه متن Computer Name پیدا خواهید کرد

وقتی شما نام کامپیوترتان را پیدا کردید می توانید آن را برای مقداردهی به خاصیت RemoteHost استفاده کنید.

اصول اتصال TCP

وقتی که یک برنامه کاربردی می سازید که از پروتکل TCP استفاده می کند، شما باید ابتدا تصمیم بگیرید که برنامه سرور خواهد بود یا کلاینت. ساختن سرور بدین معنی است که برنامه شما به پورت تعیین شده گوش خواهد داد. وقتی کلاینت درخواست اتصال می کند، سرور می تواند با درخواست موافقت کند و از آن طریق اتصال را کامل کند. هرگاه اتصال کامل شد، کلاینت و سرور می توانند آزادانه با یکدیگر در ارتباط باشند.

مراحل زیر برای ساختن یک سرور ابتدایی است:

۱- ساختن یک پروژه از نوع Standard Exe

۲- نام پیش فرض فرم را به frmServer تغییر دهید

۳- عنوان فرم را به TCP Server تغییر دهید

۴- یک کنترل WinSock را روی فرم بیندازید و نام آن را به tcpServer تغییر دهید

۵- دو TextBox روی فرم قرار دهید. نام یکی را txtSendData و یکی دیگر را txtOutput بگذارید

۶- کد زیر را به فرم اضافه کنید

```
Private Sub Form_Load()
    tcpServer.LocalPort = 1001
    tcpServer.Listen
    frmClient.Show
End Sub

Private Sub tcpServer_ConnectionRequest(ByVal requestID As Long)
```

```
    If tcpServer.State <> sckClosed Then tcpServer.Close
    tcpServer.Accept requestID
End Sub

Private Sub txtSendData_Change()
    tcpServer.SendData txtSendData.Text
End Sub

Private Sub tcpServer_DataArrival (ByVal bytesTotal As Long)
    Dim strData As String
    tcpServer.GetData strData
    txtOutput.Text = strData
End Sub
```

خاصیت LocalPort ، شماره پورت را برای تبادل داده تعیین می کند.

با اجرای متد Listen ، سرور شروع به گوش کردن به پورت می دهد. گوش کردن به معنای زیر نظر داشتن پورت برای رسیدن درخواست می باشد.

متد SendData برای ارسال داده بکار می رود.

متد Accept برای اعلان قبول داشتن بکار می رود

متد Close ، اتصال را خاتمه می دهد

رویداد `ConnectionRequest` وقتی روی می دهد که یک ماشین راه دور درخواست اتصال بکند.

رویداد `DataArrival` وقتی روی می دهد که داده جدیدی رسیده باشد.

کدهایی که در بالا آورده شده است یک برنامه ساده سرور را می سازند. اما برای کامل شده سناریو شما باید یک برنامه کلاینت را هم بسازید. مراحل زیر را برای ساختن یک برنامه کلاینت انجام دهید:

۱- یک فرم جدید به پروژه اضافه کنید و نام آن را frmClient بگذارید.

۲- عنوان فرم را به TCP Client تغییر دهید.

۳- یک کنترل WinSock به فرم اضافه کنید و نام آن را tcpClient بگذارید.

۴- به فرم جدید دو عدد TextBox اضافه کنید. نام یکی را txtSend و نام دیگری را txtOutput بگذارید.

۵- یک CommandButton روی فرم قرار دهید و نام آن را cmdConnect بگذارید.

۶- عنوان CommandButton را به Connect تغییر دهید.

۷- کد زیر را به فرم اضافه کنید.

مطمئن شوید که مقدار خاصیت RemoteHost ،که در کد مقدار آن را RemoteComputerName قرار داده ایم، را به نام دوستانه کامپیوترتان ست کرده اید.

```
Private Sub Form_Load()
    tcpClient.RemoteHost = "RemoteComputerName"
    tcpClient.RemotePort = 1001
End Sub

Private Sub cmdConnect_Click()
```

```
    tcpClient.Connect
End Sub

Private Sub txtSend_Change()
    tcpClient.SendData txtSend.Text
End Sub

Private Sub tcpClient_DataArrival (ByVal bytesTotal As Long)
    Dim strData As String
    tcpClient.GetData strData
    txtOutput.Text = strData
End Sub
```

خاصیت RemoteHost نام دوستانه ماشین راه دور است.

خاصیت RemotePort شماره پورتی است که از آن طریق می خواهیم داده ها را برای ماشین راه دور بفرستیم.

متد Connect اتصال را برقرار می کند.

متد GetData یک بلاک از داده ها را بازیابی می کند.

کدهای بالا یک برنامه کلاینت-سرور ساده را می سازد. برای استفاده از آن، برنامه را اجرا کنید و روی کلید Connect کلیک کنید. سپس متنی را در txtSendData تایپ کنید. همان متن را در txtOutput در فرم دیگر مشاهده خواهید کرد.

پذیرش بیش از یک درخواست اتصال

سرور ابتدایی که در بالا به ساخت آن اشاره شد فقط یک درخواست اتصال را می پذیرد. پذیرش بیش از یک درخواست از طریق ساختن آرایه ای از کنترلها امکانپذیر است. در این قسمت، شما لازم نیست اتصالتان را قطع کنید، اما بسادگی باید یک نمونه جدید از کنترل (و همچنین ست کردن خاصیت Index آن) بسازید، و متد Accept را در نمونه جدید فراخوانی کنید.

در کد زیر یک کنترل WinSock روی فرم داریم که نامش sckServer است، و خاصیت Index آن به 0 مقداردهی شده است، بنابراین این کنترل قسمتی از یک آرایه کنترل است. یک متغیر سطح ماژول با نام intMax تعریف کرده ایم. در رویداد Form_Load مقدار متغیر intMax را برابر 0 و خاصیت LocalPort را برای اولین کنترل در آرایه برابر1001 قرار داده ایم. سپس متد Listen کنترل را فراخوانی می کنیم. هر درخواست اتصال که برسد، ابتدا آن را برای 0 بودن Index تست می کند (مقدار کنترل گوش فرا دهنده). اگر برابر باشد، کنترل گوش فرا دهنده یکی به intMax اضافه می کند و آن را برای ساختن یک نمونه جدید از کنترل استفاده می کند. نمونه کنترل جدید ساخته شده و سپس برای توافق درخواست اتصال بکار می رود.

```
Private intMax As Long

Private Sub Form_Load()
   intMax = 0
   sckServer(0).LocalPort = 1001
   sckServer(0).Listen
End Sub
```

```
Private Sub sckServer_ConnectionRequest(Index As Integer, ByVal
requestID As Long)
   If Index = 0 Then
      intMax = intMax + 1
      Load sckServer(intMax)
      sckServer(intMax).LocalPort = 0
      sckServer(intMax).Accept requestID
      Load txtData(intMax)
   End If
End Sub
```

اصول UDP

ساختن یک برنامه مبتنی بر UDP از ساختن برنامه ای مبتنی بر TCP آسانتر است زیرا در پروتکل UDP نیازی به یک ارتباط آشکار نیست. در برنامه مبتنی بر TCP که ذکر شد، یکی از کنترلهای WinSock می بایست به حالت Listen باشد، در صورتی که کنترل دیگر باید با استفاده از متد Connect برای یک اتصال پیکربندی شود.

بطور کلی در پروتکل UDP احتیاجی به یک اتصال واضح و روشن وجود ندارد. برای ارسال داده ها بین دو کنترل سه مرحله را باید انجام دهید:

۱- خاصیت RemoteHost را برابر نام کامپیوتر دیگر قرار دهید

۲- خاصیت RemotePort را برابر مقدار خاصیت LocalPort از کنترل دیگر قرار دهید

۳- متد Bind را با LocalPort بکار برده شده فراخوانی کنید (در ادامه متد Bind توضیح داده شده است)

از آنجا که هر دو کامپیوتر می توانند در این ارتباط اهمیت یکسانی داشته باشند، می توانیم آنها را برنامه های نظیر به نظیر (Peer to Peer) بنامیم. برای نشان دادن این مسئله به کد زیر دقت کنید. این کد یک برنامه Chat می سازد که بوسیله آن دو نفر می توانند بصورت RealTime با یکدیگر صحبت کنند.

برای ساخت یک طرف برنامه UDP (Peer) مراحل زیر را انجام دهید:

۱- یک پروژه Standard Exe بسازید.

۲- نام پیش فرض فرم را به frmPeerA تغییر دهید.

۳- عنوان فرم را به Peer A تغییر دهید.

۴- یک کنترل WinSock را روی فرم قرار دهید و نام را udpPeerA بگذارید.

۵- در صفحه خواص روی خاصیت Protocol کلیک کنید و آن را به حالت UDPProtocol تغییر دهید.

۶- دو عدد TextBox روی فرم بگذارید. نام یکی را به txtSend و دیگری را به txtOutput تغییر دهید.

۷- کد زیر را به فرم اضافه کنید.

```
Private Sub Form_Load() Form Load Event
    With udpPeerA
        .RemoteHost= "PeerB"  Nothing
        .RemotePort = 1001
        .Bind 1002
    End With
    frmPeerB.Show
```

```
End Sub

Private Sub txtSend_Change()  Change Event
    udpPeerA.SendData txtSend.Text
End Sub

Private Sub udpPeerA_DataArrival (ByVal bytesTotal As
Long) Nothing
 Dim strData As String
    udpPeerA.GetData strData
    txtOutput.Text = strData
End Sub
```

توجه کنید که مقدار RemoteHost را باید برابر نام کامپیوترتان قرار دهید

برای ساخت یک طرف برنامه UDP (Peer) مراحل زیر را انجام دهید:

۱- یک فرم به پروژه اضافه کنید.

۲- نام فرم را به Peer B عوض کنید.

۳- عنوان فرم را به Peer B عوض کنید.

۴- یک کنترل WinSock روی فرم قرار دهید و نام آن را udpPeerB بگذارید.

۵- در صفحه خواص روی خاصیت Protocol کلیک کنید و گزینه UDPProtocol را انتخاب کنید.

۶- دو TextBox روی فرم قرار دهید و نام یکی را txtSend و نام دیگری را txtOutput قرار دهید.

۷- کد زیر را به فرم اضافه کنید.

```
Private Sub Form_Load() Form_Load Event
    With udpPeerB
          .RemoteHost= "PeerA"  Nothing
        .RemotePort = 1002
        .Bind 1001
    End With
End Sub

Private Sub txtSend_Change()  change event
    udpPeerB.SendData txtSend.Text
End Sub

Private Sub udpPeerB_DataArrival (ByVal bytesTotal As
Long) Nothing
    Dim strData As String
    udpPeerB.GetData strData
    txtOutput.Text = strData
End Sub
```

فراموش نکنید که مقدار خاصیت RemoteHost را برابر نام کامپیوتر خودتان قرار دهید.

برنامه را اجرا کنید و در جعبه متن txtSend در فرم خودتان متنی را تایپ کنید. متن شما در جعبه متن دیگر در فرم دیگر تایپ خواهد شد.

متد Bind:

در کد بالا نشان داده شد که وقتی می خواهید یک برنامه مبتنی بر UDP بسازید باید از متد Bind استفاده کنید. متد Bind یک پورت را برای استفاده کنترل رزرو می کند. برای مثال وقتی شما یک کنترل را با پورت شماره ۱۰۰۱ Bind کردید برنامه های دیگر نمی توانند از آن پورت استفاده

کنند و یا به آن گوش دهند. اگر شما بخواهید مانع استفاده برنامه های دیگر از ایــن پـورت شـوید، این مسئله برایتان مفید خواهد بود.

وقتی پوتکل UDP را استفاده می کنید، شما می توانیــد آزادانـه بیـن خـواص RemotePort و RemoteHost سوئیچ کنید تا وقتی که اتصال به LocalPort باقیمانده باشــد، در صورتیکـه در پروتکـل TCP شما باید قبل از تغییر RemoteHost و یا RemotePort اتصال را می بستید.