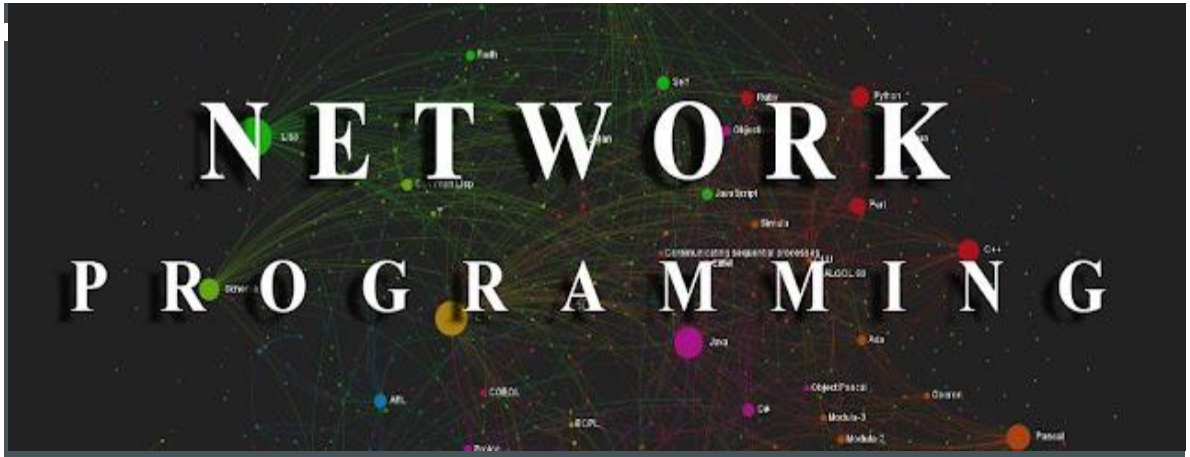


IT432 - NETWORK PROGRAMMING

DR. ALI HUSSEIN AHMED

ALI.HUSSEIN@AUN.EDU.EG

IT DEPT - OFFICE NO. 312

2022

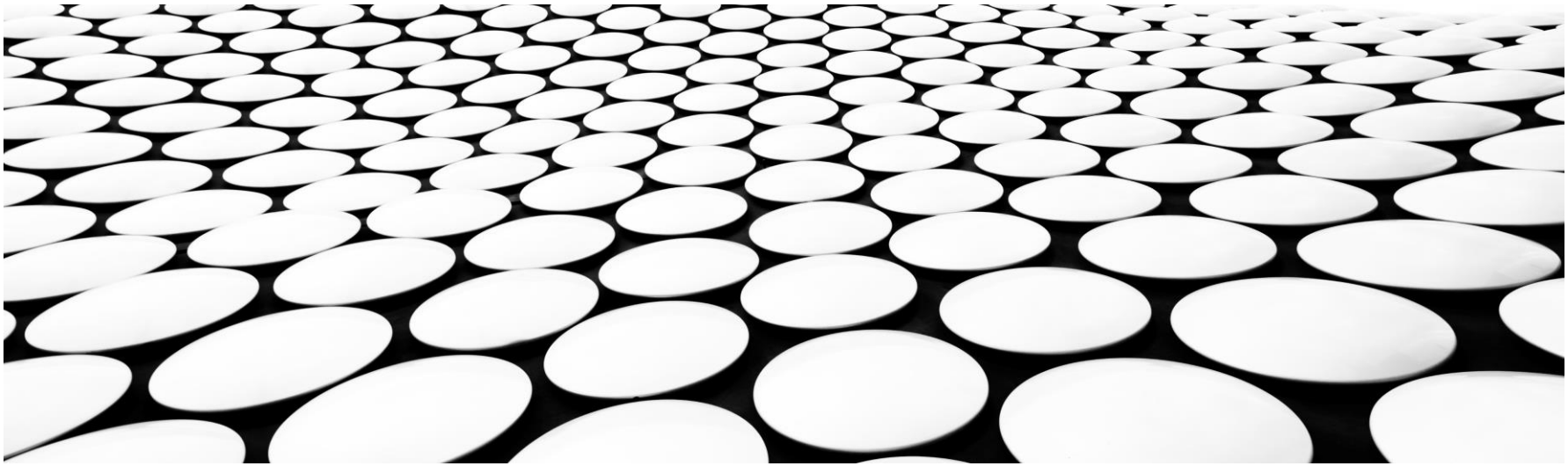
جامعة أسيوط

كلية الحاسبات والمعلومات

كلية معتمدة من الهيئة القومية لضمان الجودة والاعتماد

LECTURE 5

ASYNCHRONOUS SOCKETS



THE BLOCKING SOCKET MODE

- Sockets in blocking mode will wait forever to complete their functions, holding up other functions within the application program until they complete.
- Many programs can work quite efficiently in this mode, but for applications that work in the Windows programming environment, this can be a problem.
- Asynchronous Socket methods allow a network program to continue rather than waiting for a network operation to be performed.
- Guess which is best?

- Just as events can trigger delegates, .NET also provides a way for methods to trigger delegates.
- A delegate defines the method to be called once an event occurred.

```
checkit.Click += new EventHandler(ButtonOnClick);
```

- When the customer clicks the button, the program control moves to the ButtonOnClick() method:

```
void ButtonOnClick(object obj, EventArgs ea)
{
  results.Items.Add(data.Text);
  data.Clear();
}
```

ASYNC CALLBACK CLASS

- The .NET AsyncCallback class allows methods to start an asynchronous function and supply a delegate method to call when the asynchronous function completes.
- The Socket class utilizes the method defined in the AsyncCallback to allow network functions to operate asynchronously in background processing.
- It signals the OS when the network functions have completed and passes program control to the AsyncCallback method to finish the network function.

- The Socket asynchronous methods split common network programming functions into two pieces:
  - A **Begin** method that starts the network function and registers the AsyncCallback method
  - An **End** method that completes the function when the AsyncCallback method is called

.NET ASYNCHRONOUS SOCKET METHODS

Requests Started By…	Description of Request	Requests Ended BY…
BeginAccept()	To accept an incoming connection	EndAccept()
BeginConnect()	To connect to a remote host	EndConnect()
BeginReceive()	To retrieve data from a socket	EndReceive()
BeginReceiveFrom()	To retrieve data from a specific remote host	EndReceiveFrom()
BeginSend()	To send data from a socket	EndSend()
BeginSendTo()	To send data to a specific remote host	EndSendTo()

THE BEGINACCEPT() AND ENDACCEPT() METHODS

```
Socket sock = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
sock.Bind(iep);
sock.Listen(5);
sock.BeginAccept(new AsyncCallback(CallAccept), sock);
```

```
private static void CallAccept(IAsyncResult iar)
{
    Socket server = (Socket)iar.AsyncState;
    Socket client = server.EndAccept(iar);
}
```

THE BEGINCONNECT() AND ENDCONNECT() METHODS

```
Socket newsock = new Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 9050);
newsock.BeginConnect(iep, new AsyncCallback(Connected), newsock);
```

```
public static void Connected(IAsyncResult iar)
{
    Socket sock = (Socket)iar.AsyncState;
    try
    {
        sock.EndConnect(iar);
    }
    catch (SocketException)
    {
        Console.WriteLine("Unable to connect to host");
    }
}
```

SENDING AND RECEIVING DATA

```
sock.BeginSend(data, 0, data.Length, SocketFlags.None,
 new AsyncCallback(SendData), sock);


private static void SendData(IAsyncResult iar)
{
    Socket server = (Socket)iar.AsyncState;
    int sent = server.EndSend(iar);
}
```

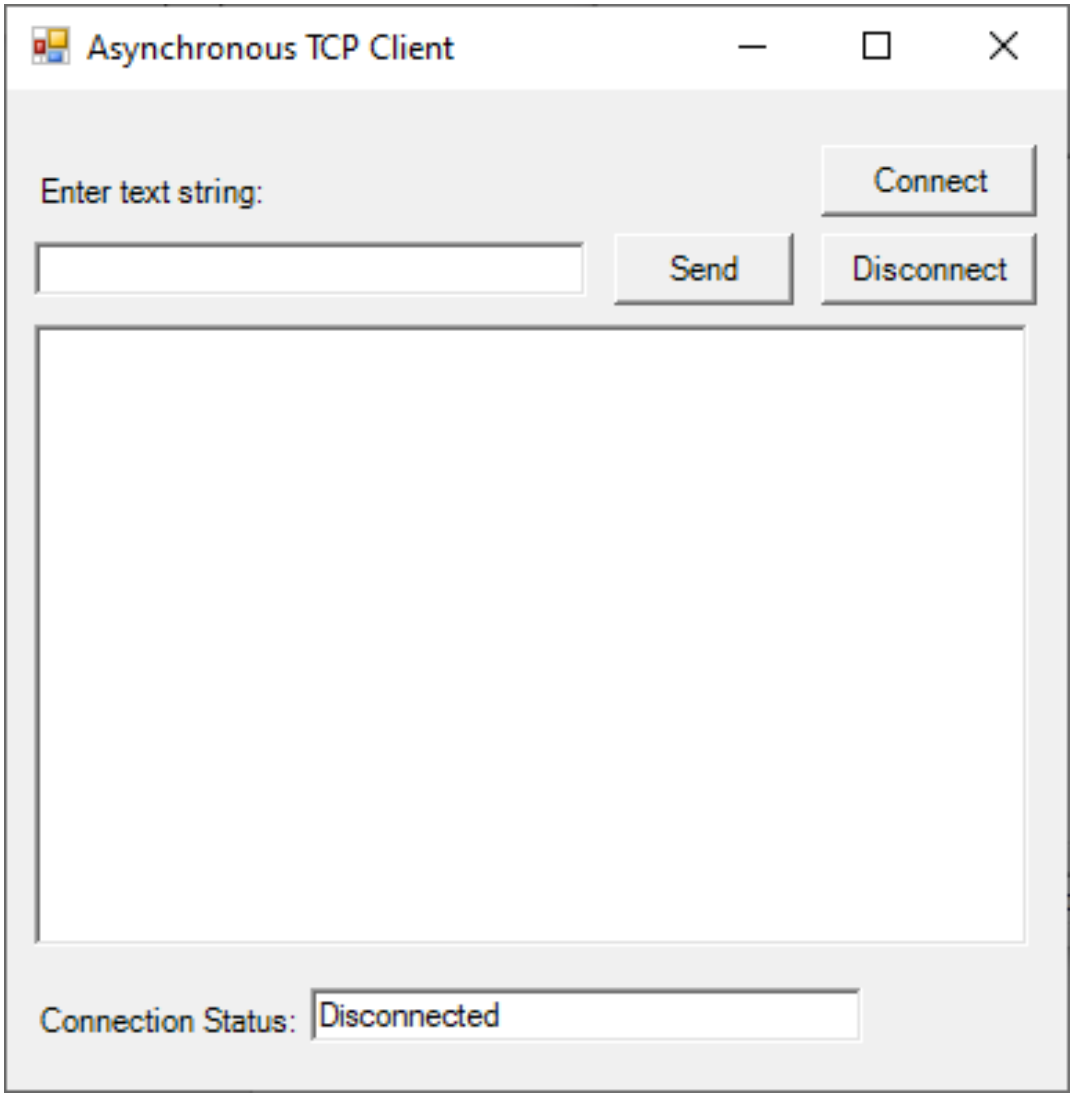
OTHER ASYNCH CALLS

- The BeginSendTo() and EndSendTo() Methods
- The BeginReceive() and EndReceive() Methods
- The BeginReceiveFrom() and EndReceiveFrom() Methods

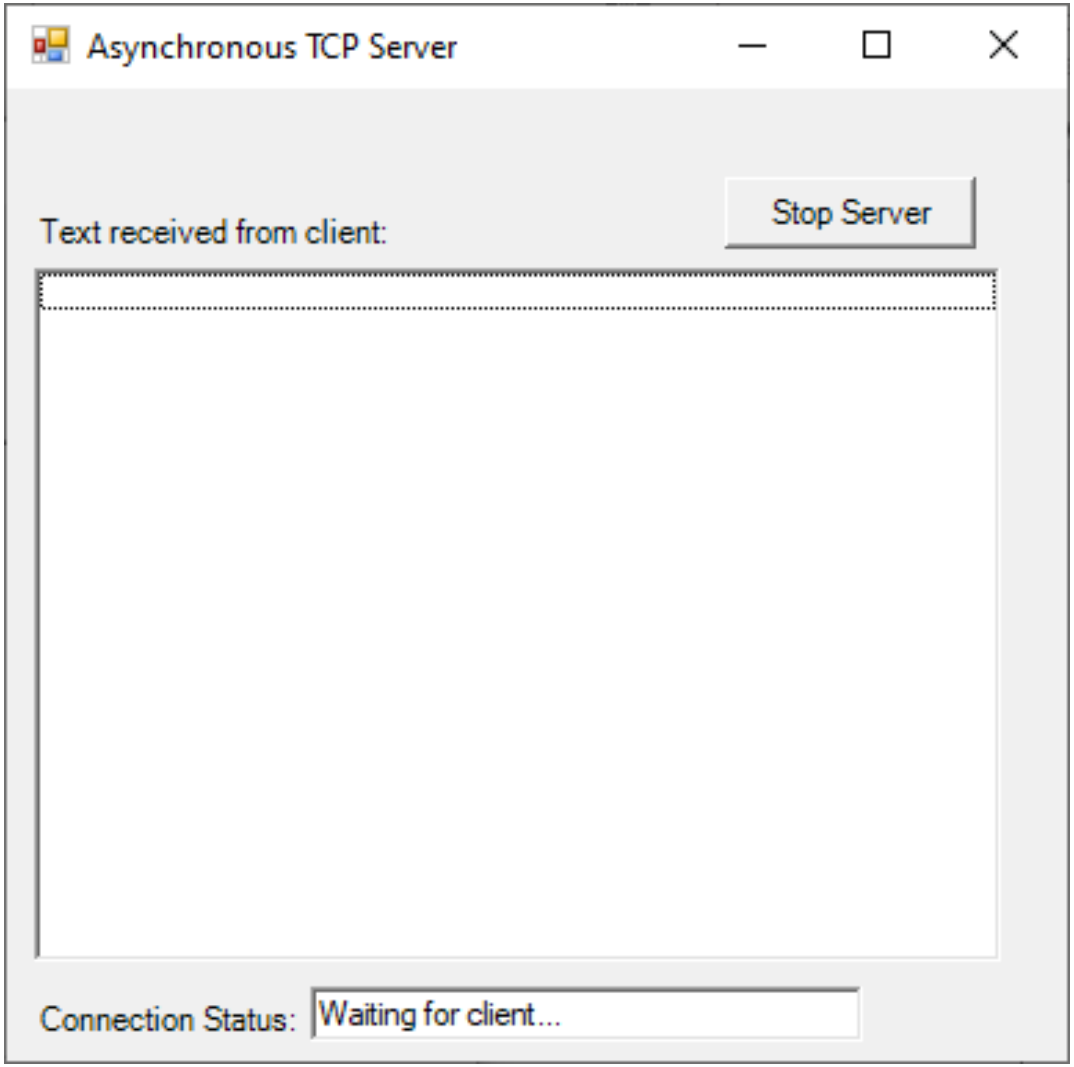
EXAMPLE

- In the next few slides, we will create a windows application for both clint and server.
- We will notice that our GUI did not freeze during the blocking socket calls.

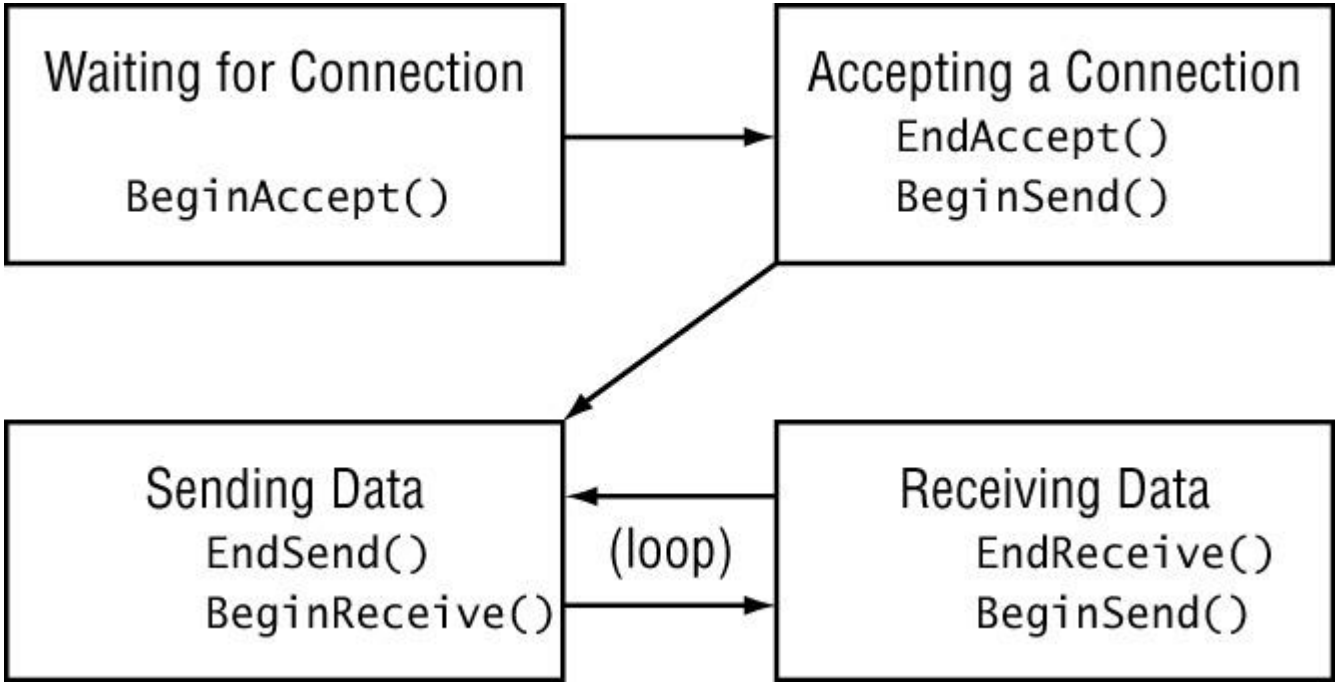
THE CLIENT PROGRAM



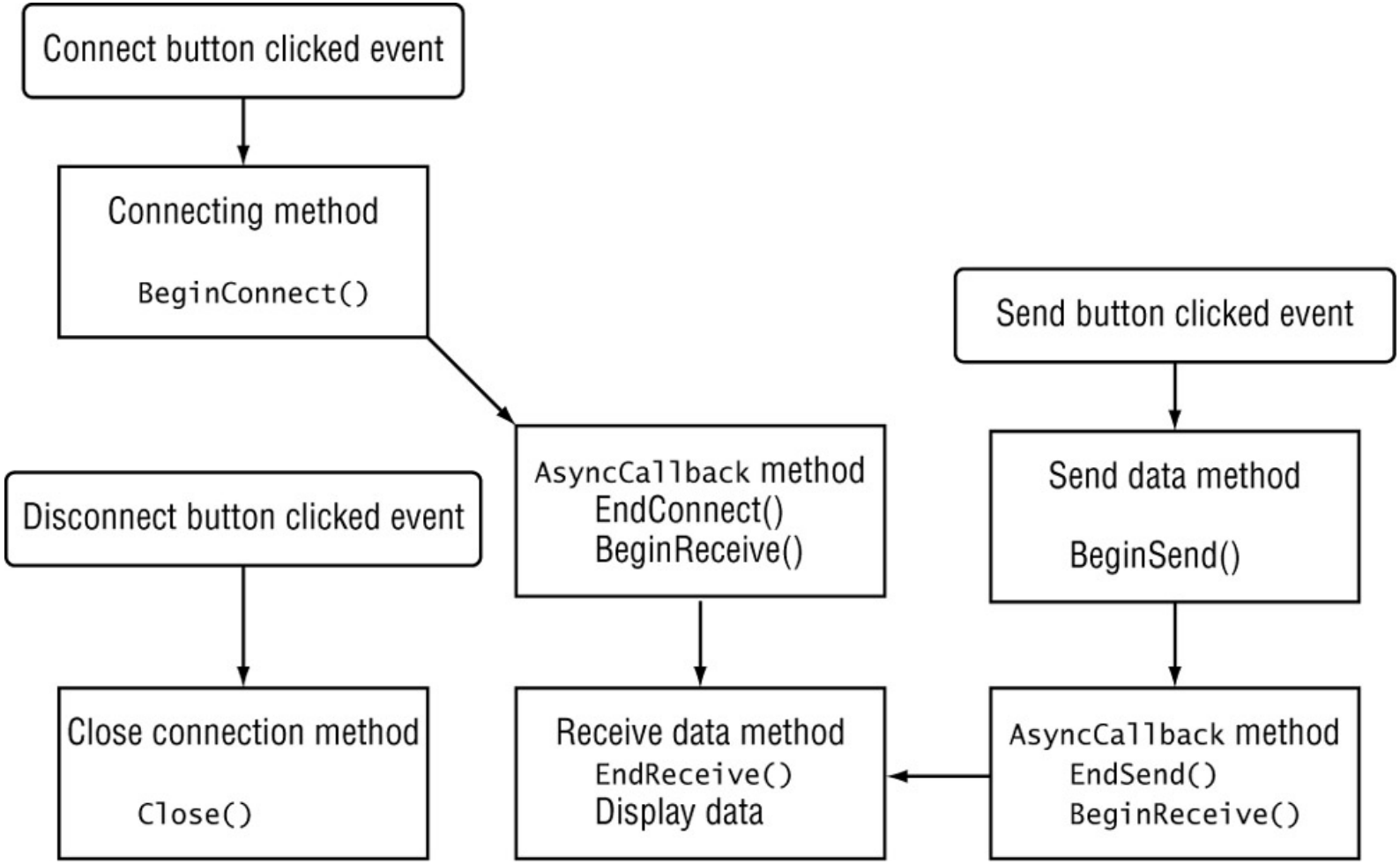
THE SERVER PROGRAM



THE SERVER LOOP



THE CLIENT


Connect button clicked event
Connecting method
BeginConnect()
Send button clicked event
AsyncCallback method
EndConnect()
BeginReceive()
Send data method
BeginSend()
Disconnect button clicked event
Close connection method
Close()
Receive data method
EndReceive()
Display data
AsyncCallback method
EndSend()
BeginReceive()

THE SOURCE CODE

- Client.cs
- Server.cs

NEXT TOPIC

- Multithreaded server