Milestone 1: Basic Remoting Infrastructure

Implement a Remoting prototype using the chat application shown in class. This prototype should serve as the basis of the Remoting infrastructure of your Poker application.

- Start with the chat client (a Windows.Forms application) and add a new menu to configure the chat application.

![Chat Application Image]

- The menu item “Host Chat Room” is used to initialize a Remoting service. That is, if this menu item is selected, the application registers a Remoting factory object that implements the following interface:

```java
public interface IFactory
{
    IBroadcaster GetBroadcasterInstance();
    IBroadcaster GetBroadcasterInstance( MessageArrivedHandler Handler );
}
```

Furthermore, when hosting a chat room, your application must also start listing for HostLocator queries (i.e., UpdHostLocator.StartListener()).

To facilitate the Remoting setup, you should use the BasicRemoting library.

- The menu item “Join” is used to join a chat room. To join a chat room, you need to acquire a remote factory object and use it the get a broadcaster object. Therefore, you need to scan for possible hosts using the UpdHostLocator.QueryHosts method. That is, you should define a Scan dialog that has to start a query for servers as shown below.
The Scan dialog defines three elements: a ListBox, a ProgressBar, and a Button. The progress starts at 0 and ends at 10 and advances every second (i.e., as a result of the timer1’s “Tick” event).

Furthermore, upon the calling the Scan dialog’s ShowDialog method, you should be set the button1’s “Text” property to “Cancel”, the “DialogResult” property to DialogResult.Cancel, and the scan form’s “CancelButton” property to “button1”. After 10 seconds the dialog has to stop listening for incoming responses and indicate this by changing the button1’s “Text” property to “Ok”, the “DialogResult” property to DialogResult.OK, and the scan form’s “AcceptButton” property to “button1”. By clicking on an element in the ListBox, you can select the desired remote host.

Note, if you host a chat room, you also need to join it immediately.

- Use the following class to turn off the lifetime management mark all Remoting object immortal:

  ```csharp
  public override object InitializeLifetimeService()
  {
      return null;
  }
  ```

  Note that if a Windows.Form object defines an event handler that is registered with a remote object, then the lifetime of this very same Windows.Form object is also controlled by the Remoting infrastructure.
Notes:

- You cannot register a HTTP-channel twice. That is, if you host a chat room, you register a HTTP-channel that listens at port 9000. The same channel is used when you want to send messages to other chatters.
- Once you made your decision to host or to join a chat room, disable the menu items.
- Even if you host a chat room, you need to acquire a Remoting factory object. That is, your application will act as a client.
- Define the server startup in a separate class.
- Reuse everything. The original chat room solution is available online. However, you are required to keep copyright notices in the code.
- This milestone should help you to tackle problems that you may encounter when using the .NET Remoting infrastructure. You will be able to use everything in the final Poker application.
- When you close the application do not forget to shut down the HostLocator. Otherwise, your application will remain active.

Submission deadline: Monday, October 31, 2003, 4:10 p.m.

Submission procedure: on paper in class (.cs files only) and electronically using the milestone1 script, which is located in ~cs430x/public/bin. Please use the printout of the submission confirmation email as cover page and check the problems that you have solved.

In order to submit your homework solutions, go (using your CS UNIX account) into the directory that contains your solution (i.e., C#-source files and all related project files). In that directory run the command “~cs430x/public/bin/milestone1”. After a successful submission, your will receive a confirmation email. Before the due date, you can resubmit your solutions as often as you like.

On the department's Windows XP systems you can use the command csc to compile C#-programs. However, it is recommended to use Visual Studio .NET, because most assignments require some GUI work.