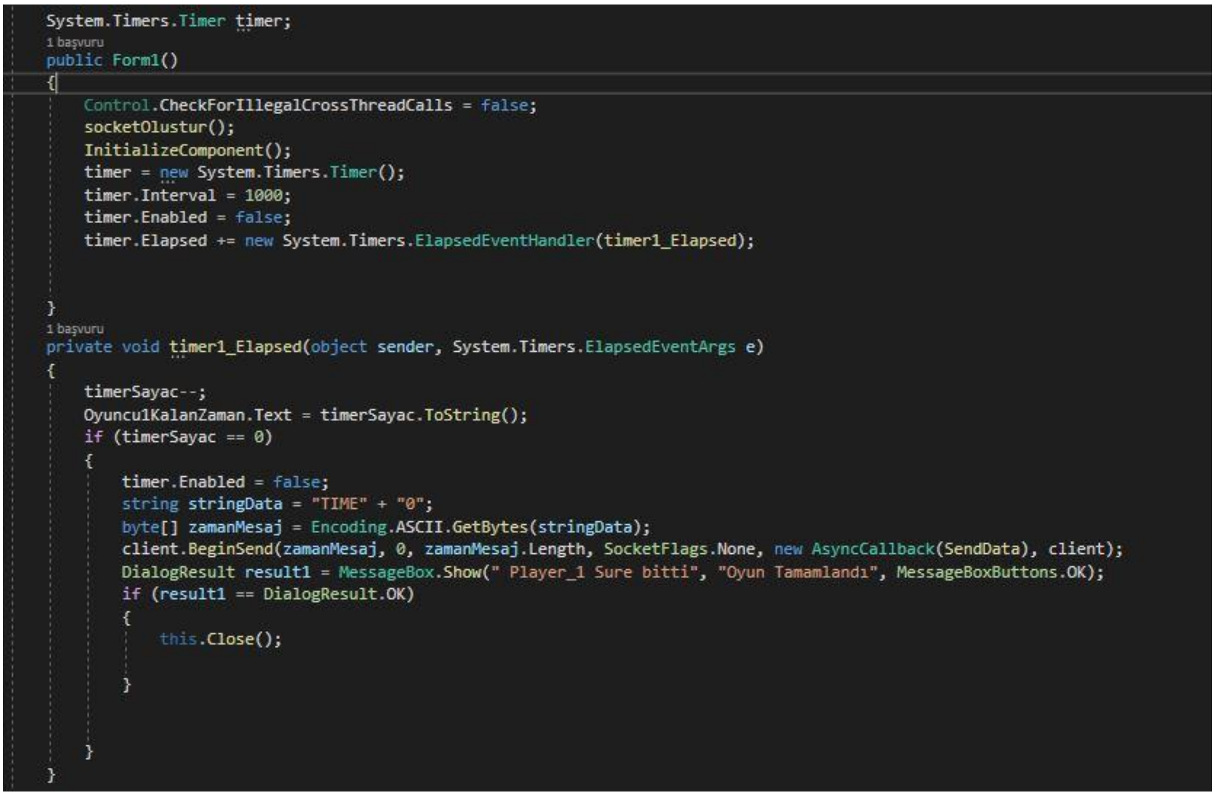
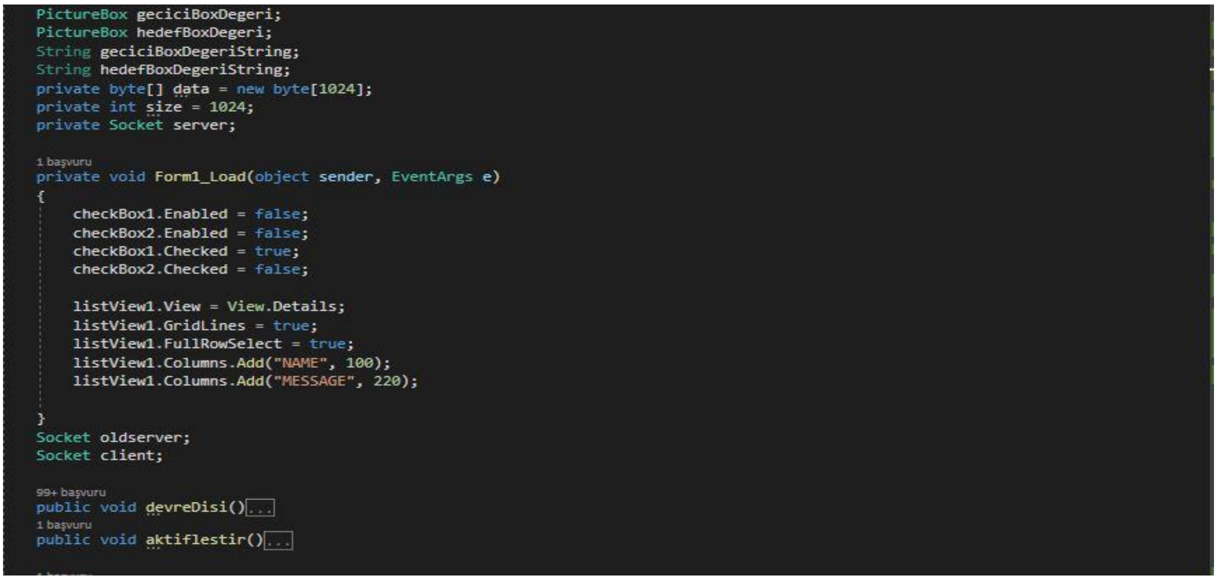
PLAYER_1 – SERVER DOCUMENTATION



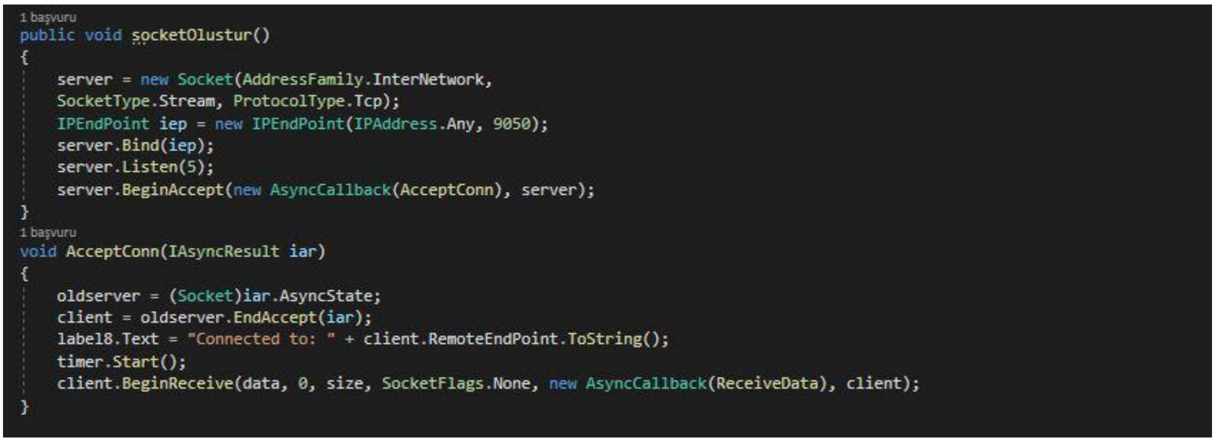
```
System.Timers.Timer timer;
1 başvuru
public Form1()
{
    Control.CheckForIllegalCrossThreadCalls = false;
    socketOlustur();
    InitializeComponent();
    timer = new System.Timers.Timer();
    timer.Interval = 1000;
    timer.Enabled = false;
    timer.Elapsed += new System.Timers.ElapsedEventHandler(timer1_Elapsed);


}
1 başvuru
private void timer1_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
{
    timerSayac--;
    Oyuncu1KalanZaman.Text = timerSayac.ToString();
    if (timerSayac == 0)
    {
        timer.Enabled = false;
        string stringData = "TIME" + "0";
        byte[] zamanMesaj = Encoding.ASCII.GetBytes(stringData);
        client.BeginSend(zamanMesaj, 0, zamanMesaj.Length, SocketFlags.None, new AsyncCallback(SendData), client);
        DialogResult result1 = MessageBox.Show(" Player_1 Sure bitti", "Oyun Tamamlandı", MessageBoxButtons.OK);
        if (result1 == DialogResult.OK)
        {
            this.Close();

        }


    }
}
```

This is the first part we will see when we open the project. In the Form1 method, there are definitions for "Timer" that calculates the player time when the project is started for the first time. Also, the function that will provide the TCP connection is called. The Interval property determines how many milliseconds the "Timer" will run. Also, the Enabled property is set to false so that it does not work on the first start. The timer1_Elapsed method runs every 1000 milliseconds and prints it on the screen by reducing the time. Also, if the duration of Player_1 is 0, it notifies in the message box and sends this information to Player_2 and the game ends.



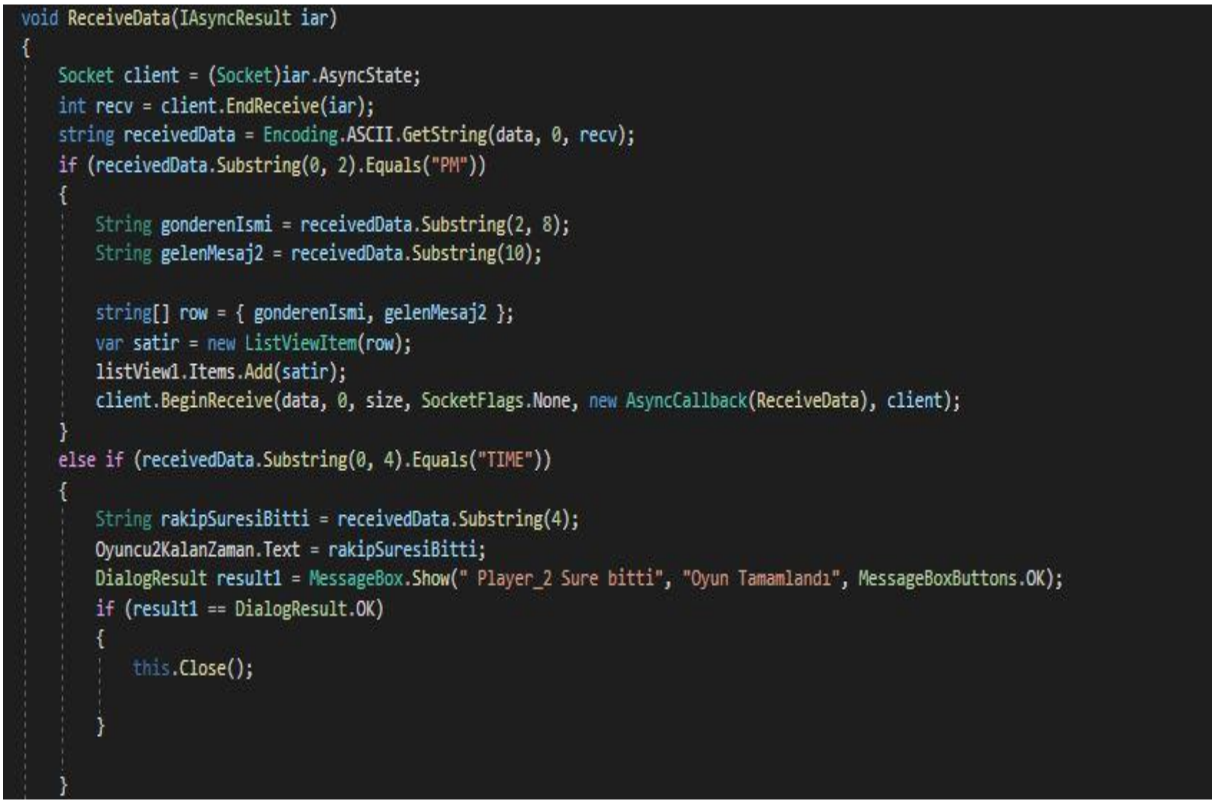
```
PictureBox geciciBoxDegeri;
PictureBox hedefBoxDegeri;
String geciciBoxDegeriString;
String hedefBoxDegeriString;
private byte[] data = new byte[1024];
private int size = 1024;
private Socket server;

1 başvuru
private void Form1_Load(object sender, EventArgs e)
{
    checkBox1.Enabled = false;
    checkBox2.Enabled = false;
    checkBox1.Checked = true;
    checkBox2.Checked = false;

    listView1.View = View.Details;
    listView1.GridLines = true;
    listView1.FullRowSelect = true;
    listView1.Columns.Add("NAME", 100);
    listView1.Columns.Add("MESSAGE", 220);

}
Socket oldserver;
Socket client;

99+ başvuru
public void devreDisi()...
1 başvuru
public void aktiflestir()...
```

When the project is run for the first time, the ListView created for the chat screen is defined and the CheckBoxes are set for Player_1 and Player_2, which will show who is next. The "aktiflestir" and "devreDisi" methods enable or disable the clicking feature of PictureBoxes created to represent each square on the board. This prevents the 2nd move from being played after each move.

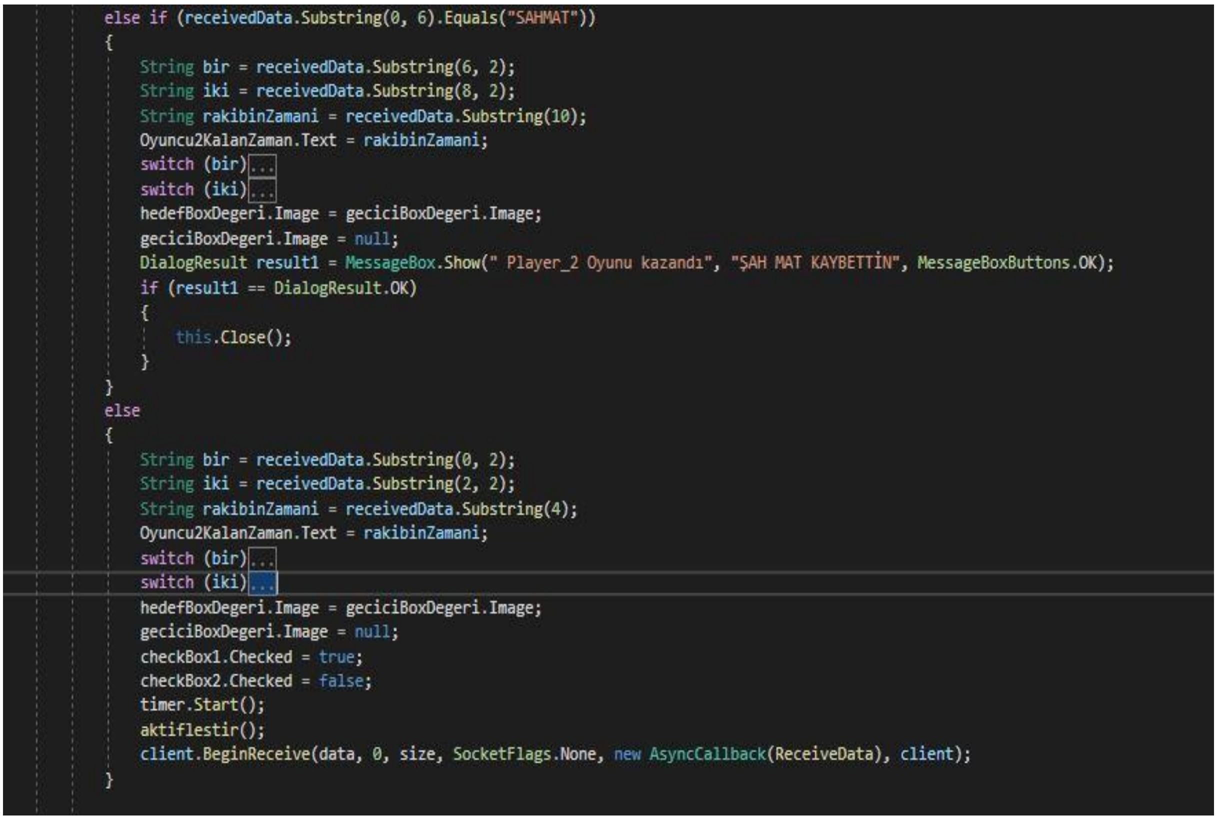


```
1 başvuru
public void socketOlustur()
{
    server = new Socket(AddressFamily.InterNetwork,
    SocketType.Stream, ProtocolType.Tcp);
    IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
    server.Bind(iep);
    server.Listen(5);
    server.BeginAccept(new AsyncCallback(AcceptConn), server);
}
1 başvuru
void AcceptConn(IAsyncResult iar)
{
    oldserver = (Socket)iar.AsyncState;
    client = oldserver.EndAccept(iar);
    label8.Text = "Connected to: " + client.RemoteEndPoint.ToString();
    timer.Start();
    client.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), client);
}
```

The "socketOlustur" method is the server creation code for our TCP connection. We say that we are ready for everyone with the "Bind" feature. Also, thanks to List(5), we can queue up 5 customers. Thanks to the so-called "BeginAccept", when someone connects to us, we show the connected ip address on the screen. and the timer starts running. In addition, it remains asynchronously ready for a message that may come from the chat section. Thanks to async, we avoid GUI blocking while waiting for the message.Also if you want to play on two different IPs, this can be done by replacing "IPAdress.Any" with the original IP.



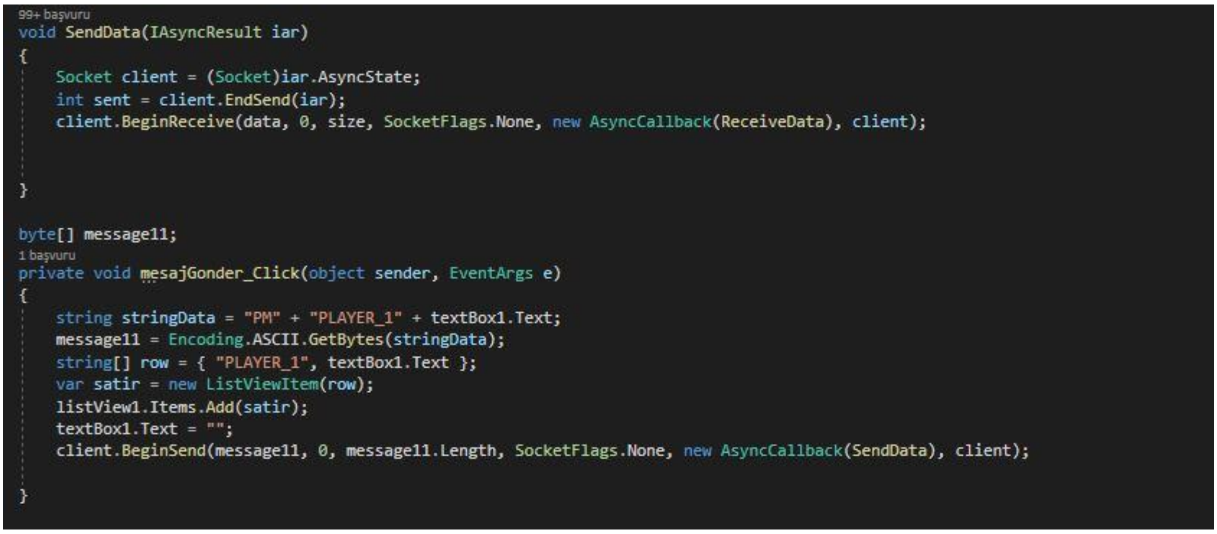
```
void ReceiveData(IAsyncResult iar)
{
    Socket client = (Socket)iar.AsyncState;
    int recv = client.EndReceive(iar);
    string receivedData = Encoding.ASCII.GetString(data, 0, recv);
    if (receivedData.Substring(0, 2).Equals("PM"))
    {
        String gonderenIsmi = receivedData.Substring(2, 8);
        String gelenMesaj2 = receivedData.Substring(10);

        string[] row = { gonderenIsmi, gelenMesaj2 };
        var satir = new ListViewItem(row);
        listView1.Items.Add(satir);
        client.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), client);
    }
    else if (receivedData.Substring(0, 4).Equals("TIME"))
    {
        String rakipSuresiBitti = receivedData.Substring(4);
        Oyuncu2KalanZaman.Text = rakipSuresiBitti;
        DialogResult result1 = MessageBox.Show(" Player_2 Sure bitti", "Oyun Tamamlandı", MessageBoxButtons.OK);
        if (result1 == DialogResult.OK)
        {
            this.Close();

        }

    }
```



```
else if (receivedData.Substring(0, 6).Equals("SAHMAT"))
{
    String bir = receivedData.Substring(6, 2);
    String iki = receivedData.Substring(8, 2);
    String rakibinZamani = receivedData.Substring(10);
    Oyuncu2KalanZaman.Text = rakibinZamani;
    switch (bir)...
    switch (iki)...
    hedefBoxDegeri.Image = geciciBoxDegeri.Image;
    geciciBoxDegeri.Image = null;
    DialogResult result1 = MessageBox.Show(" Player_2 Oyunu kazandı", "ŞAH MAT KAYBETTİN", MessageBoxButtons.OK);
    if (result1 == DialogResult.OK)
    {
        this.Close();
    }
}
else
{
    String bir = receivedData.Substring(0, 2);
    String iki = receivedData.Substring(2, 2);
    String rakibinZamani = receivedData.Substring(4);
    Oyuncu2KalanZaman.Text = rakibinZamani;
    switch (bir)...
    switch (iki)...
    hedefBoxDegeri.Image = geciciBoxDegeri.Image;
    geciciBoxDegeri.Image = null;
    checkBox1.Checked = true;
    checkBox2.Checked = false;
    timer.Start();
    aktiflestir();
    client.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), client);
}
```

ReceiveData method consists of 4 parts.

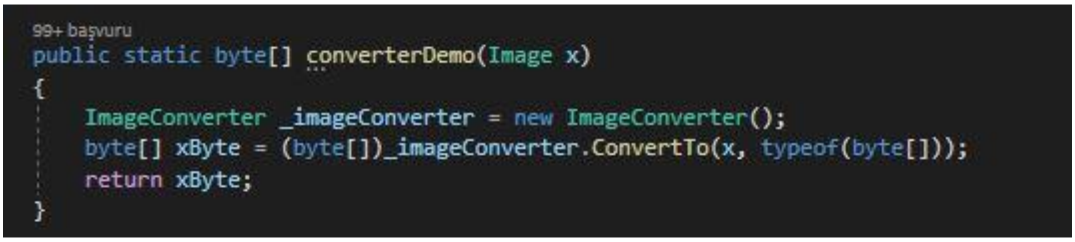
- First of all, the byteArray that comes asynchronously is assigned to a new array. Then the string value is retrieved using this array. If it starts with "PM", which is the starting tag I use to identify chat messages, first the player name (Player_2) and then the message is imported and added to the Listview by using the subString method. And because the message or move can come again, the RecieveData method is called asynchronously.
- If the opposing player's time runs out, he sends his time using the "TIME" tag. And we get this, we understand that the game is over and we show it in the messageBox.
- If the opposing player eats our king, he sends us a message with "SAHMAT" tag that the game is over. We show this message and the move on the screen and finish the game. By using "Switch-case", we take the conjugate of the opponent's move and show it on the screen. In this way, we can also obtain the "From -To" information. The position of the piece selected as "bir" is the position it plays, selected as "iki"."geciciBoxDegeri" is the position of the selected piece, and "hedefBoxDeğeri" is the position where the move is made.
- The last part is to get move information in a normal way. After taking the opponent's move and showing it on the screen, our screen functions are activated thanks to the "aktifleştir" method and it is our turn. Also, our time starts, the opponent's stops. And the checkBox is exchanged, which shows that it is our turn.



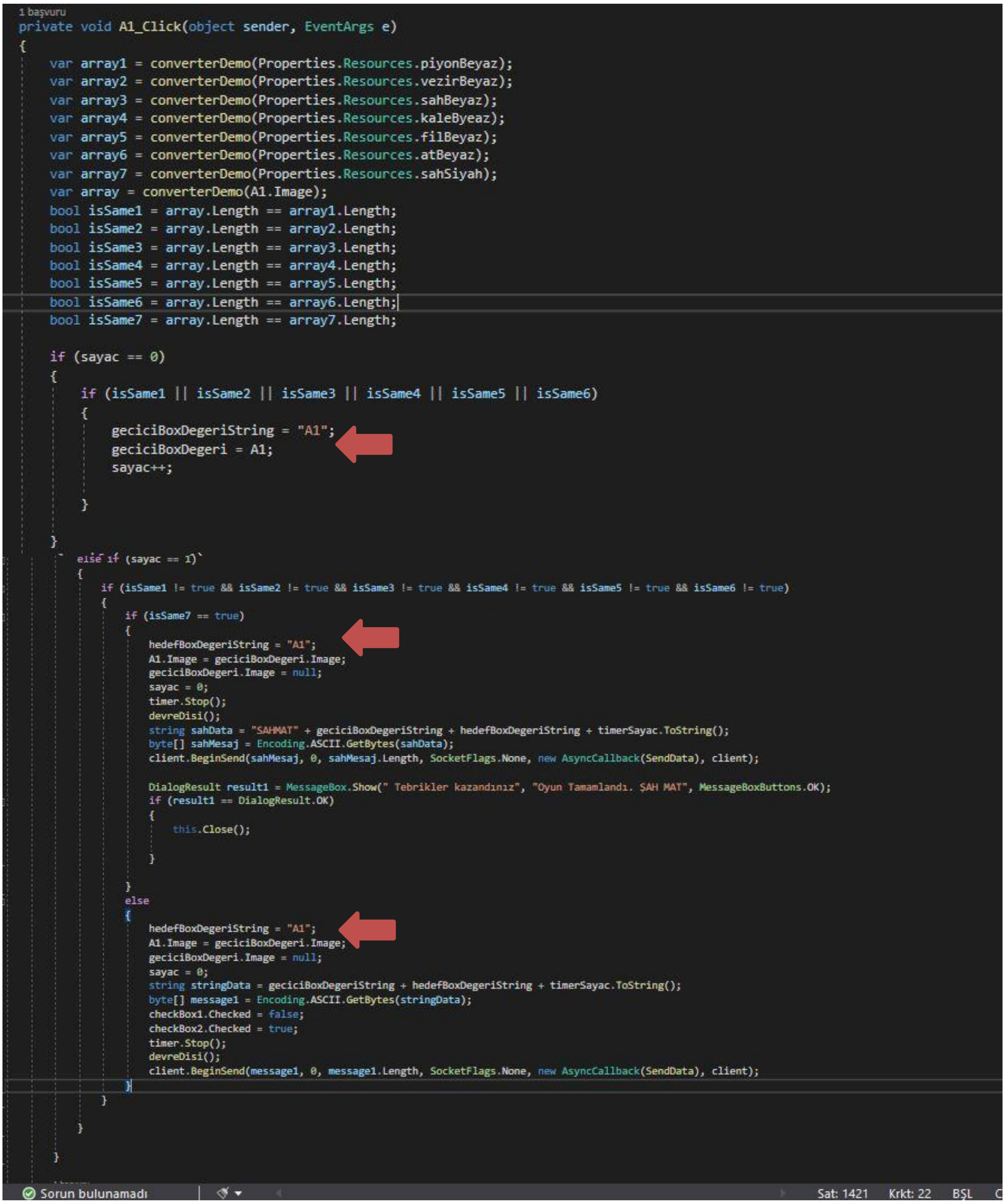
```
99+ başvuru
void SendData(IAsyncResult iar)
{
    Socket client = (Socket)iar.AsyncState;
    int sent = client.EndSend(iar);
    client.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), client);


}

byte[] message11;
1 başvuru
private void mesajGonder_Click(object sender, EventArgs e)
{
    string stringData = "PM" + "PLAYER_1" + textBox1.Text;
    message11 = Encoding.ASCII.GetBytes(stringData);
    string[] row = { "PLAYER_1", textBox1.Text };
    var satir = new ListViewItem(row);
    listView1.Items.Add(satir);
    textBox1.Text = "";
    client.BeginSend(message11, 0, message11.Length, SocketFlags.None, new AsyncCallback(SendData), client);

}
```

The "SendData" method is responsible for sending the data sent as a parameter as it is.

When the send message button is pressed, the "PM" tag and the player name and message content are assigned to the string. Then it is added to our Listview screen and sent by giving a parameter to the SendData method.



```
99+ başvuru
public static byte[] converterDemo(Image x)
{
    ImageConverter _imageConverter = new ImageConverter();
    byte[] xByte = (byte[])_imageConverter.ConvertTo(x, typeof(byte[]));
    return xByte;
}
```



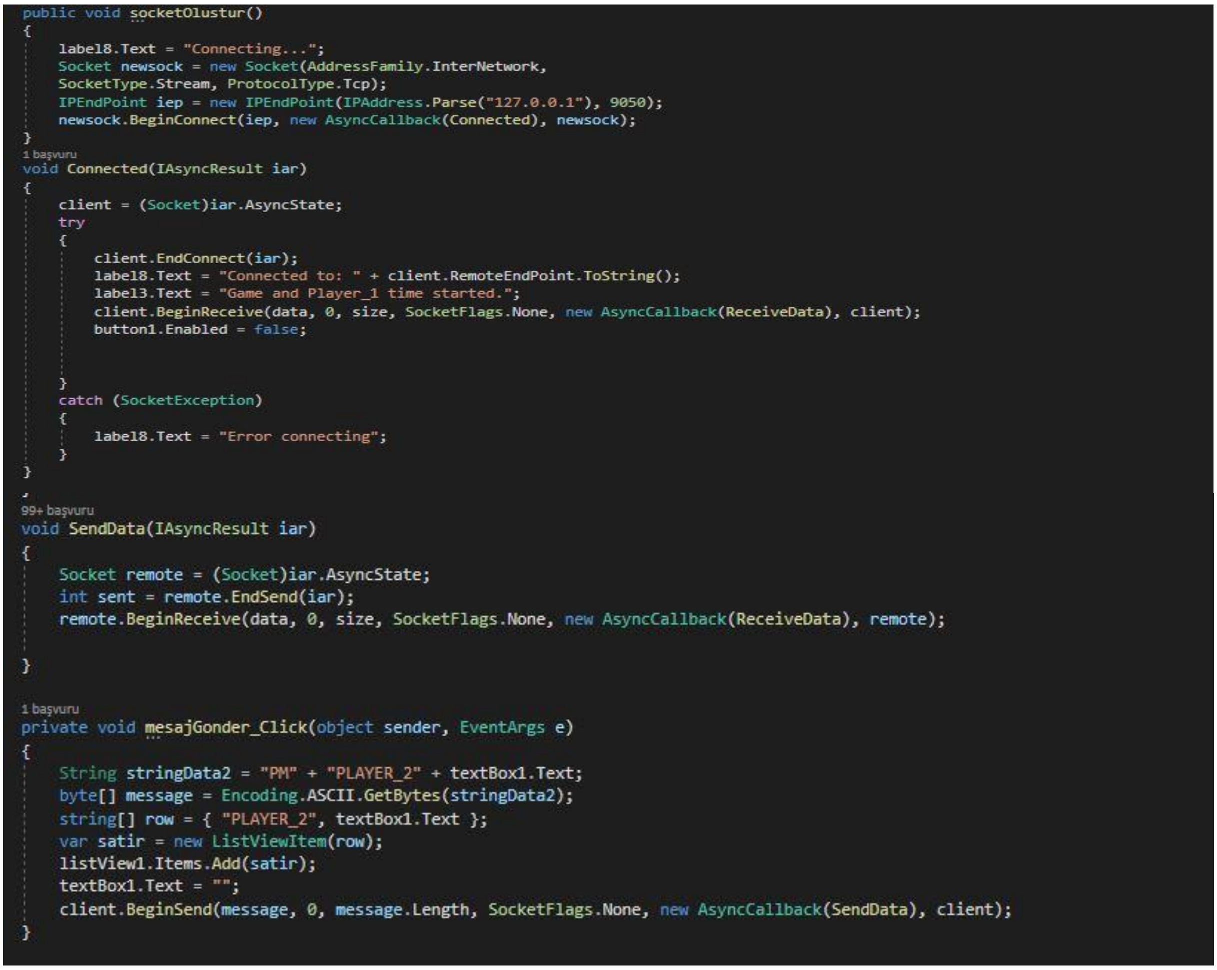
```
private void A1_Click(object sender, EventArgs e)
{
    var array1 = converterDemo(Properties.Resources.piyonBeyaz);
    var array2 = converterDemo(Properties.Resources.vezirBeyaz);
    var array3 = converterDemo(Properties.Resources.sahBeyaz);
    var array4 = converterDemo(Properties.Resources.kaleByeaz);
    var array5 = converterDemo(Properties.Resources.filBeyaz);
    var array6 = converterDemo(Properties.Resources.atBeyaz);
    var array7 = converterDemo(Properties.Resources.sahSiyah);
    var array = converterDemo(A1.Image);
    bool isSame1 = array.Length == array1.Length;
    bool isSame2 = array.Length == array2.Length;
    bool isSame3 = array.Length == array3.Length;
    bool isSame4 = array.Length == array4.Length;
    bool isSame5 = array.Length == array5.Length;
    bool isSame6 = array.Length == array6.Length;
    bool isSame7 = array.Length == array7.Length;

    if (sayac == 0)
    {
        if (isSame1 || isSame2 || isSame3 || isSame4 || isSame5 || isSame6)
        {
            geciciBoxDegeriString = "A1";
            geciciBoxDegeri = A1;
            sayac++;

        }

    }
    else if (sayac == 1)
    {
        if (isSame1 != true && isSame2 != true && isSame3 != true && isSame4 != true && isSame5 != true && isSame6 != true)
        {
            if (isSame7 == true)
            {
                hedefBoxDegeriString = "A1";
                A1.Image = geciciBoxDegeri.Image;
                geciciBoxDegeri.Image = null;
                sayac = 0;
                timer.Stop();
                devreDisi();
                string sahData = "SAHMAT" + geciciBoxDegeriString + hedefBoxDegeriString + timerSayac.ToString();
                byte[] sahMesaj = Encoding.ASCII.GetBytes(sahData);
                client.BeginSend(sahMesaj, 0, sahMesaj.Length, SocketFlags.None, new AsyncCallback(SendData), client);

                DialogResult result1 = MessageBox.Show(" Tebrikler kazandınız", "Oyun Tamamlandı. ŞAH MAT", MessageBoxButtons.OK);
                if (result1 == DialogResult.OK)
                {
                    this.Close();

                }

            }
            else
            {
                hedefBoxDegeriString = "A1";
                A1.Image = geciciBoxDegeri.Image;
                geciciBoxDegeri.Image = null;
                sayac = 0;
                string stringData = geciciBoxDegeriString + hedefBoxDegeriString + timerSayac.ToString();
                byte[] message1 = Encoding.ASCII.GetBytes(stringData);
                checkBox1.Checked = false;
                checkBox2.Checked = true;
                timer.Stop();
                devreDisi();
                client.BeginSend(message1, 0, message1.Length, SocketFlags.None, new AsyncCallback(SendData), client);
            }
        }

    }

}
```

This method is the click method of the pictureBox. It is the same for all pictureBoxes, only the naming part of the pictureBox that I show with the arrow changes. So I will only show this as an example.

- We cannot compare pictureBox images directly in c#. Therefore, we need to convert all white (Player_1) stones and black king stones added to the project into byte array using the converterDemo method.

- We count the number of clicks using the counter variable we defined above. If the counter is 0 and the pictureBox picture we click is the white stone, we choose the stone we clicked to play. And we assign it to the “geciciBoxDegeri”.
- If the counter value is 1, that is, if we have chosen a stone to play before, we check the pictureBox image that we clicked as the target so that it is not a white stone. If it is not a white piece , we move the piece at that position to this position using the “geciciBoxDeğeri “ and delete the image in the “geciciBoxDeğeri “ . Then the timer stops and we are prevented from making the 2nd move thanks to the “devreDisi” method. CheckBox is exchanged and we send our move information and remaining time to the opponent.
- If the pictureBox we selected as the target contains a black king, the same procedure is repeated, but the information that the game is over is transmitted to the other party using the "SAHMAT" tag.

PLAYER_2 – CLIENT DOCUMENTATION



```
public void socketOlustur()
{
    label8.Text = "Connecting...";
    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);
}
1 başvuru
void Connected(IAsyncResult iar)
{
    client = (Socket)iar.AsyncState;
    try
    {
        client.EndConnect(iar);
        label8.Text = "Connected to: " + client.RemoteEndPoint.ToString();
        label3.Text = "Game and Player_1 time started.";
        client.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), client);
        button1.Enabled = false;


    }
    catch (SocketException)
    {
        label8.Text = "Error connecting";
    }
}

99+ başvuru
void SendData(IAsyncResult iar)
{
    Socket remote = (Socket)iar.AsyncState;
    int sent = remote.EndSend(iar);
    remote.BeginReceive(data, 0, size, SocketFlags.None, new AsyncCallback(ReceiveData), remote);

}

1 başvuru
private void mesajGonder_Click(object sender, EventArgs e)
{
    String stringData2 = "PM" + "PLAYER_2" + textBox1.Text;
    byte[] message = Encoding.ASCII.GetBytes(stringData2);
    string[] row = { "PLAYER_2", textBox1.Text };
    var satir = new ListViewItem(row);
    listView1.Items.Add(satir);
    textBox1.Text = "";
    client.BeginSend(message, 0, message.Length, SocketFlags.None, new AsyncCallback(SendData), client);
}
```

The only difference from the client(Player_2) server is that the client object and try-catch are used when creating a socket. This is because the application does not close in case of a possible connection error. At the same time, the remote object is used when sending data.