T.C.

MANİSA CELAL BAYAR ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

CWG-CinsFlags War Game

Final Homework

170316045 Oğuzhan ÇEVİK

Teacher

Assoc.Prof.Dr. Muhammed Cinsdikici

MANİSA 2022

CONTENTS

Page

1 INTRODUCTION....3

1.1 What is the mission?....3
1.2 What did I do?....3

2 WHAT DOES SERVER DO....4

2.1 Setup Process....4
2.2 Encryption Process for Send Flag Numbers to Client....4
2.3 Initialize Server Flags....5
2.4 Act According to Click Operation....5
2.5 sendNumber Function....7
2.6 Win or Lose....7

3 WHAT DOES CLIENT DO....8

3.1 Setup Process....8
3.2 Decryption Process....8
3.3 Initialization....10
3.4 First Player Logic....11
3.5 Act According to Click Operation....12
3.6 sendNumber Function....12

4 BONUS PART:GOLDEN FLAG....13

4.1 What is Golden Flag....13
4.2 Coding Section....13

5 LEARNING FROM MISTAKES....15

5.1 Label and Button....15
5.2 Thread Error....15
5.3 String and Char Problem....15
5.4 Possible Points....15
5.5 C# Knowledge....15

6 SCREENSHOTS FROM THE GAME 16

6.1 Golden Flag Round 16
6.2 Disappear on Normal Round 17
6.3 Win/Lose 19

7 REFERENCES 21

1. INTRODUCTION

1.1 What is the mission?

1: The game will be played between two PCs (Two users - Server/Client architecture).

2: Both players will sew 5 own flags on their own copies of the same map (seen below).

3: Flag coordinates will not be shared between players.

4: A flag represents a certain area. (Example: area of 15 pixels x 15 pixels.)

5: Both players will try to find their respective flags. Finds the flag of the opposing player will be the owner/manager of that region.

6: The positions of the flags will remain fixed during the game

1.2 What did I do?

1: The game will be played between two PCs (Two users - Server/Client architecture). (✔)

2: Both players will see 5 own flags on their own copies of the same map (**They see 28 flag and in this 28 flag, they also see their 5 colored(red or blue) flag in their own copies**) (✔)

3: Flag coordinates will not be shared between players. (✔)

4: A flag represents a certain area. (**My flags represents 20x20 area**) (✔)

5: Both players will try to find their respective flags. Finds the flag of the opposing player will be the owner/manager of that region. (✔)

6: The positions of the flags will remain fixed during the game(✔)

2. WHAT DOES SERVER DO?

2.1 Setup Process

Server selects 10 **unique** random flags for server and client. And it saves server's cities to myCity List and saves client's cities to opponentCity List. Server also opens array called valuesForCity. This array contains integer 2 as many as flag numbers. This array solves the problem called "Which flag is mine , Which flag is yours". In this array;

Integer 1 means: It is opponent's city.

Integer 0 means: It is my city.

Integer 2 means: It is nobody's flag.

At the first, every element on this array is integer 2. According to myCity and opponentCity arrays, server makes the index of valuesForCity 0 or 1. So Server can handle the "which flag is mine and which flag is yours" problem according to these values of indexes.

For example:

myCity=[1,3,5,6,7] opponentCity=[9,11,4,8,2]

valuesForCity[1] will be 0 because it is in myCity list.

valuesForCity[9] will be 1 because it is in opponenyCity list.

valuesForCity[14] will be 2 because it is nobody's flag.

After these operation server will be waiting for the connection that will come from client side.

2.2 Encryption Process for Send Flag Numbers to Client

Server has information about server's flags and client's flags on code section. When the connection comes, it converts them into string with a format. That format is;

For example:

myCity=[1,3,5,6,7] opponentCity=[9,11,4,8,2]

strMyCity="1,3,5,6,7"

strOpponentCity="9,11,4,8,2"

strToSend="9,11,4,8,2/1,3,5,6,7"

And with that, when server sends that string, Client will know that first part is client's flags and the other part is server's flags and it will convert them into integer numbers according to this information.

2.3 Initialize Server's Flags

Server has disableAll function and it makes all flags disable. This function calls by server at the first. Because Client starts to game as a first player. When client makes his/her decision, he/she clicks to button and server get that information. Server also has activateAll function that makes all flags (which is not server's) active. When server gets information from client, it makes his own flags blue and disabled them for the one time only at the beginning of the game. Because these flags are fixed during game. After that, It calls activateAll function to make the other flags clickable(which is not server's).

2.4 Act according to Click Operation

Server takes information about which flag that client chose. It is number in string format so when client sends that information, we convert it into integer and use it in our code. Server calls opponentClick function with this information. And these performClick functions works according to number and if I have that flag then calls that performClick function but if I do not have that flag then server make that flag disapper.

Example: index is number that comes from client side. The reason that I am making that button enable is inside of in this first "if", according to condition it is my flag.

```
if (index == 20 && button20.Visible == true && myCity.Contains(index))
{
    button20.Enabled = true;
    button20.PerformClick();


}
else if (index == 20 && button20.Visible == true && !myCity.Contains(index))
{
    button20.Visible = false;

}
```

So what is performClick? performClick is function that works when we click that spesific button.

Example:

As I said index will be 1 if it is opponent's flag and it will be 0 if it is my flag.

So, If it is my flag that means opponent clicked it because I can not click my own flag it is disabled to me. So valuesForCity[20]==0 condition will be working on, it will make flag20's "enable condition" false and make it red to show that opponent got your flag.And also increase opponent score by 1. Client and Server has point initialized with 0. If Server gets Client's flag,server gets the point and make that flag blue. If Client's gets Server's flag, Client gets the point and make that flag red. When one of them reached the 5 point(because 10 flags in this game seperated 5-5) game will be over and all flags will be disabled.

Note:If someone captures opponent flag, that flag can not be taken back during game. I decided that because of game duration. I wanted this game to be short.

```
private void button20_Click(object sender, EventArgs e)
{
    if (valuesForCity[20] == 1)
    {
        button20.Enabled = false;
        button20.BackColor = Color.Blue;
        myScore++;
        if (myScore == 5)
        {
            status.Text = "YOU WIN !";
            disableAll();
        }
        myScore1.Text = myScore.ToString();
        sendNumber("20");
    }
    if (valuesForCity[20] == 0)
    {
        button20.Enabled = false;
        button20.BackColor = Color.Red;
        opponentScore++;
        if (opponentScore == 5)
        {
            status.Text = "YOU LOSE !";
            disableAll();
        }
        textBox1.Text = opponentScore.ToString();
    }
```

That functions also will be working on when i clicked that flag too so if valuesForCity[20]==2 then it will be disapear and it will send that number as a string via sendNumber Function to client. So client will make that flag disappear too.

2.5 sendNumber Function

This function allows us to send that "which flag has been chosen". Every click function use that function in some situation. If i click nobody's flag it works, or if i click opponent's flag this function works to send that information(number in string format) to client.

```
//SENDING CLIKED BUTTON INDEX AS A STRING TO CLIENT
public void sendNumber(String x)
{
    //IF I DID NOT WIN GAME WAITCURSOR ENABLE
    if (myScore != 5)
    {
        UseWaitCursor = true;
        disableAll();
    }


    byte[] message1 = Encoding.ASCII.GetBytes(x);
    client.BeginSend(message1, 0, message1.Length, SocketFlags.None,
                new AsyncCallback(SendData), client);

}
```

This function also does that if i cliked a flag but i did not win so that means it will be opponent's turn so all the flags on server side should be disabled and cursor must be waitCursor to show that "it is not your turn". Then it sends it to client via Tcp protocol.

2.6 Win or Lose

Every time I click a flag, if it is mine flag, i check opponent's points to end this game. If it is 5 then i can disableAll and write on screen "YOU LOSE" but if it is opponent's flag then i check myPoints to end this game. If it is 5 then i can disableAll and write on screen "YOU WIN". It works the same way as client. Client has same the function as server has.

3. WHAT DOES CLIENT DO?

3.1 Setup Process

After the server has opened, we can open client part of the game. When we opened the client part, client loads flags in string format via Tcp Protocol(makes tcp connection at the beginning).Then opens valuesForCity array as server(Same type 0 is mine 1 is opponent 2 is nobody's flag).

Flag data comes from server is "1,3,4,6,7/2,8,9,23,21"(this is an example for format)

3.2 Decryption Process

In this part we have to make flag data usable. If the data comes from server is string, then we have to convert into integer then save it to list. This process is;

```
//SEPERATING INDEXES
string[] subs = receivedData.Split('/');
foreach(var str in subs)
{
    Console.WriteLine(str);
}
 i = 0;
//SEPERATING CITIES AND CONVERTING INTO INTEGER AND SAVING INTO LISTS
foreach (var str in subs)
{
    if (i == 0)
    {
        string[] withoutComma = str.Split(',');
        foreach (var letter in withoutComma)
        {
            int number = Int32.Parse(letter);
            myCity.Add(number);
        }
    }
    else
    {
        string[] withoutComma = str.Split(',');
        foreach (var letter in withoutComma)
        {
            int number = Int32.Parse(letter);
            opponentCity.Add(number);
        }
    }
    i++;
}
```

With "subs" we split the flags, first element of subs is "1,3,4,6,7" and second is "2,8,9,23,21".

We are selecting each element with for loop. If i=0 then we are doing some process on first

element that means myCity. We split them convert from “1,3,4,6,7” to “13467”. Then we use foreach for that part too. We are converting every number in string format to integer format and save it to myCity list. We are doing same process when i!=0 that means opponentCity(server cities).

3.3 Initialization

When we get the ours cities and opponent’s cities we can change valuesForCity array with those informations. Example:

```
//DETERMINATION OF MY CITIES
for (int j = 0; j < myCity.Count; j++)
{
    valuesForCity[myCity[j]] = 0;

}
//DETERMINATION OF OPPONENT CITIES
for (int j = 0; j < opponentCity.Count; j++)
{
    valuesForCity[opponentCity[j]] = 1;

}
```

After that, we can color our flags as Red. For example for index 0(flag 0). We are doing this process for every flag.

```
// DECLARES RED TEAMS FLAGS
for (int j = 0; j < valuesForCity.Length; j++)
{
    if (j == 0 && valuesForCity[j] == 0)
    {
        button0.Enabled = false;
        button0.BackColor = Color.Red;
    }
    if (j == 1 && valuesForCity[j] == 0)
    {
        button1.Enabled = false;
        button1.BackColor = Color.Red;
    }
```

3.4 First Player Logic

We are the first player as a client, it is like White in Chess. So we need to make a move, i mean click a flag. This process is:

```
//IF I GOT ENEMY FLAG I'M MAKING THAT FLAG MINE
if (valuesForCity[0] == 1)
{
    button0.Enabled = false;
    button0.BackColor = Color.Red;
    myScore++;
    if (myScore == 5)
    {
        //FINISHING GAME
        status.Text = "YOU WIN !";
        disableAll();
    }
    //WRITING SCORE
    myScore1.Text = myScore.ToString();
    //SENDING INDEX TO USE TO SERVER
    sendNumber("0");
}
else if (valuesForCity[0] == 0)
{
    //IF CITY IS MINE AND ENEMY CLICKED IT, THEN I AM MAKING THAT FLAG BLUE
    //AND GIVING ENEMY +1 POINTS
    button0.Enabled = false;
    button0.BackColor = Color.Blue;
    opponentScore++;
    if (opponentScore == 5)
    {
        //FINISHING GAME
        status.Text = "YOU LOSE !";
        disableAll();
    }
```

If valuesForCity[selectedFlag] ==0 then it means it is opponent's flag then we have to color that red and increase our score by 1. Then send it to server to say that "Hey I got your area".

If valuesForCity[selectedFlag] ==1 then It is my area and i can not click that, only opponent can click that area so i have to color that blue in my map too. I do not sendNumber function in here because if somene gets the flag, that flag can not be clickable rest of the game. I do not have to send it back to server, Because i did not click it.

If valuesForCity[selectedFlag] ==2 then it is nobody's so we can make sure that flag disappears in our map too. Then we can send it to server side to say "Hey i clicked empty flag" so server can update his map too.

3.5 Act according to Click Operation

```
//WHEN OPPONENT CLICKED ONE FLAG I AM TAKING INDEX AND MAKING CLICK OPERATION ON MY FLAGS
public void opponentClick(int index)
{
    //IF I HAVE THAT CITY I HAVE TO MAKE ENABLE FIRST
    if (index == 0 && button0.Visible == true && myCity.Contains(index))
    {
        button0.Enabled = true;
        button0.PerformClick();

    }
    //IF I DO NOT HAVE THAT CITY THEN I HAVE TO REMOVE THAT BUTTON
    else if (index == 0 && button0.Visible == true && !myCity.Contains(index))
    {
        button0.Visible = false;

    }
```

It works the same way as server. When server sends number that he/she clicked, we can call performClick function if it is still visible and if i have that flag in myCity List. If i do not have that flag then we make that disappear.

3.6 disableAll, enableAll and waitCursor

disableAll function makes all flags(buttons) disable. We use it for turn(round). If it is my turn then we can call enableAll function to make all flags(which i do not have) enable, and disable the waitCursor element. If it is not then we can call disableAll functions to disable all buttons and enable waitCursor element.

3.7 sendNumber Function

It works the same way as server. It takes clicked flag number as string and sends it to server side. If myscore is not equal to 5(it means i did not win yet) then i can make my flags disable and waitCursor enabled.

```
//SENDING CLIKED BUTTON INDEX AS A STRING TO SERVER
public void sendNumber(String x)
{
    //IF I DID NOT WIN GAME WAITCURSOR ENABLE
    if (myScore != 5)
    {
        UseWaitCursor = true;
        disableAll();
    }
    byte[] message = Encoding.ASCII.GetBytes(x);

    client.BeginSend(message, 0, message.Length, SocketFlags.None,
                new AsyncCallback(SendData), client);
}
```


4. BONUS PART: GOLDEN FLAG

4.1 What is Golden Flag?

There will be 10 flag in every game. 5 flags are opponent's and 5 flags are my flags. I thought that if there was another flag which would finish the game. This is the **GOLDEN FLAG**. We have a counter called Round and it was initialized by 0. It will increase by 1 when someone clicks the button(flag). We have another variable called GoldenFlagRound. It is for select the random round for GoldenFlag. If it is 9, that means Golden Flag randomly appears in round 9, and if you find it you can win the game at one shot. Golden Flag appears in random round and at random location. It is different for 2 opponent. For example server may have the Golden Flag in round 10 at random flag which is not one of the 10 special flags. Client may have also Golden Flag in round 7 at random flag which is not one of the 10 special flags. When Golden Flag round comes textbox will say "GOLDEN FLAG ROUND". If they can not find the Golden Flag then it will be opponent's turn, it will be like some another round. Golden Flag is initialized by 999. When we get the Golden Flag round then it will be some random Flag, for example flag 24 will be the Golden Flag if we can not find the flag 24, then we are going to lost our chance.

After we missed it, game will initialize Golden Flag as 999 again. Because there will be no Golden Flag for the person Who played this round.

4.2 Code Section

Random a = new Random()

GoldenFlagRound = a.Next(6, 11); // It will generate random number between 6-10. It is same code for client and server.

```
if (Round == GoldenFlagRound)
{
    status.Text = "GOLDEN FLAG ROUND";
    foreach (var city in valuesForCity)
    {
        GoldenFlag = a.Next(0, valuesForCity.Length);
        if (valuesForCity[GoldenFlag] == 2)
        {
            if (GoldenFlag == 0 && button0.Visible)
            {
                Console.WriteLine("button 0 has Golden Flag");
                break;
            }
            if (GoldenFlag == 1 && button1.Visible)
            {
                Console.WriteLine("button 1 has Golden Flag");
                break;
            }
```

Golden Flag must be nobody's flag,after we check that, we are trying to find which flag is random flag and we are printing it on the console to check.

```
else if(Round>GoldenFlagRound)
{
    status.Text = "";
    GoldenFlag = 999;
}
Console.WriteLine("GOLDEN FLAG {0}",GoldenFlag);
```

If we miss the Golden Flag then we initialize Golden Flag as 999 again and textbox must be empty string, there should not be writing "GOLDEN FLAG ROUND".

We are checking every flag for Golden Flag. If that is Golden Flag then we are setting the score as 5. And make the Back Color as Golden, disable all buttons(flags) and send to other side "finish" string. If it is not Golden Flag we are doing same thing, it is also nobody's flag then we make it disapper and send it to other side as number(in string format) .

```
if (GoldenFlag == 1)
{
    myScore = 5;
    myScore1.Text = "5";
    status.Text = "YOU WIN, YOU FOUND THE GOLDEN FLAG!";
    button1.BackColor = Color.Gold;
    disableAll();
    sendNumber("finish");

}
else
{
    sendNumber("1");
    button1.Visible = false;
}
```

When we get the "finish" string we know that it is ended by Golden Flag. So we do;

```
if (receivedData == "finish")
{
    opponentScore = 5;
    textBox1.Text = "5";
    status.Text = "YOU LOSE, OPPONENT FOUND THE GOLDEN FLAG";
    disableAll();

}
```

5. LEARNING FROM MISTAKES

5.1 Label and Button

I decided to make flags “label”. But label and buttons can have only 1 parent so i can not give every button or label to same parent. If i can not give as parent. When i click to label, label moves different position. So i decided to change that to button. I got the same error. When i searched for it, I realized image was the problem. It was not fit. I set the sizeMode as StrechedImage and my problem was solved.

5.2 Thread Error

I do not know why, but I was getting error about Thread When Tcp worked on. When i searched for it I found that code on [1]https://www.gencayyildiz.com/blog/c-capraz-is-parcacigi-islemi-gecerli-degil-hatasi-ve-cozumu/ .

```
Control.CheckForIllegalCrossThreadCalls = false;
```

It solved my problem.

5.3 String and Char Problem

When i tried to send the flag numbers as a string like “1234578905”. On the other side, when i want to get first element I got an error like “it is an char that you want to get it so it must be char array not string array” I changed it but error was the same. So I figured out that if i send them like “1,2,3,4,5/,6,7,8,9,12” it is not problem. It solved my problem with “split” function.

5.4 Possible Points

First, I designed this game with 74 buttons. It was different to code but I wanted to do that if game starts, flag must be random too in one country. For example:

In moskovia it was 6 flags but one of them should be main flag for this country and it must be in the game. Then i realized it was bad and so long to find out which flag is whom. After that, I decided to do this game, 28 flag and 28 city.

5.5 C# knowledge

I do not have coding experience in C# so much. I was just a beginner so I looked for some examples on [2]https://docs.microsoft.com/tr-tr/dotnet/csharp/ .

6. SCREENSHOTS FROM THE GAME

6.1 Golden Flag Round

Server

Client

Server

Client

6.2 Disappear on Normal Round

Client(Before Click)

Server(Before Client Clicks)

Client(After Click)

Server(After Client Clicks)

6.3 Win/Lose

Server

Client

As you can see on the client side, server captured 3 area from me. I must have 2 are left but I captured 5 more area and won so at the end I have 7 area on my own.

7. REFERENCES

[1] https://www.gencayyildiz.com/blog/c-capraz-is-parcacigi-islemi-gecerli-degil-hatasi-ve-cozumu/

[2] https://docs.microsoft.com/tr-tr/dotnet/csharp/

[3]C# Network Programming by Richard Blum (TCP codes)