Nu te supăra frate

19.01.2024

Îndrumător:

dr. ing. Daniel Morariu

Student:

Mateoi Paul-Andrei-Dumitrel

223/1

Istoric Versiuni

Data	Versiune	Descriere	Autor
6.12.2023	v1	-Clasa Zar functionala -Interfata grafica creeata -Obiecte pe tabla creeate (pioni, zar)	Mateoi Paul-Andrei-Dumitrel
9.12.2023	v2	-Jocul este functional, pionii se deplasează pe tablă	Mateoi Paul-Andrei-Dumitrel
20.12.2023	v3	-Clasele au fost refăcute, am sters clasele initiale JucatorRosu si JucatorAlbastru, și am creeat clasele Jucator si Joc	Mateoi Paul-Andrei-Dumitrel
12.01.2024	v4	-Retea functională	Mateoi Paul-Andrei-Dumitrel
14.01.2024	v5	-Implementare casuta de selectare a tare a IP-ului -Imbunatatiri la nivelul logicii jocului	Mateoi Paul-Andrei-Dumitrel, internet

Cuprins

1 Specificarea cerințelor software

1.1 Introducere

Proiectul vizează crearea unei versiuni digitale a jocului de societate clasic "Nu te supăra frate", adaptată pentru mediul online. Scopul este de a permite jucătorilor să interacționeze și să se bucure de acest joc având posibilitatea de a se juca de pe două dispozitive. Jocul a fost dezvoltat pentru a suporta multiplayer în rețea locală, oferind o interfață prietenoasă și reguli intuitive, asemănătoare cu cele ale jocului de masă tradițional.

1.1.1 Obiective

- Dezvoltarea Interfeței Utilizatorului: Crearea unei interfețe grafice prietenoase și intuitive, care să faciliteze interacțiunea cu jocul.
- Implementarea Logicii de Joc: Codificarea regulilor de bază ale jocului "Nu te supăra frate", cum ar fi mișcarea pieselor și aruncarea zarurilor plus alte reguli specifice:
    - Posibilitatea ca pionii să poată intra pe tablă doar după ce jucătorul primește cifra 6 când dă cu zarul.
    - Posibilitatea de a da iar cu zarul în cazul în care primește cifra 6.
    - Posibilitatea de a intra în casă după parcurgerea drumului și de a câștiga jocul.
    - Posibilitatea jucatorului de a da cu zarul sau de a muta piese doar când este rândul său.
- Conectivitatea în Rețea Locală: Implementarea funcționalităților necesare pentru a permite ca doi jucători să se conecteze și să joace împreună într-o rețea locală.
- Utilizarea principiilor OOP cum ar fi encapsularea, moștenirea, polimorfismul în structurarea și dezvoltarea codului.

1.1.2 Definiţii, Acronime şi Abrevieri

Pentru a denumi variabilele în cadrul proiectului în mare parte am folosit convenția CamelCase, adică începerea numelui cu literă mică pentru primul cuvânt, iar apoi literă mare pentru cuvintele următoare.

In cazul claselor am folosit convenția PascalCase, adică începerea tuturor cuvintelor din nume cu litere mari.

Switch-case - structură de control în programare utilizată pentru a executa diferite acțiuni bazate pe diferite condiții. Este o alternativă eficientă la o serie lungă de instrucțiuni if-else, în special când există multe condiții de verificat. Switch case selectează execuția unui bloc de cod bazat pe valoarea unei variabile sau a unei expresii.

Lista - structură de date dinamica, parte a bibliotecii de clase de bază (Base Class Library), care poate stoca o colecție ordonată de elemente.

1.1.3 Tehnologiile utilizate

- Visual Studio 2022 (Forms Application .NET Framework 4.7.2) cu urmatoarele extensii:
    - GitHub Copilot
    - JetBrains ReSharper
    - Class Designer
- Microsoft Office Word 2021
- GitHub
- OpenAI ChatGPT

- VisualParadigm Online

1.2 Cerințe specifice

- Conectivitatea în rețea locală.
- Interfață grafică prietenoasă cu utilizatorul si ușor de utilizat.
- Mutarea pionilor de tabla de joc
- Posibilitatea de a intra în casă după parcurgerea drumului și de a câștiga jocul.
- Posibilitatea jucatorului de a da cu zarul sau de a muta piese doar când este rândul său.
- Posibilitatea ca pionii să poată intra pe tablă doar după ce jucătorul primește cifra 6 când dă cu zarul.

2 Mutarea pionilor pe tabla de joc

2.1 Descriere

Funcționalitatea mutării pionilor este una de bază pentru acest joc, deoarece fară aceasta jocul nu s-ar putea desfățura. Pionii se pot muta pe tabla doar după anumite reguli, cum ar fi intrarea pe tablă doar dupa ce jucătorul primește cifra 6 de la zar, sau posibilitatea de a intra in zona de final după parcurgerea drumului.

2.2 Fluxul de evenimente

2.2.1 Fluxul de bază

Implementarea acestei funcționalități a fost realizată începând din clasa Jucator prin metoda mutaPion ce primește ca parametri culoarea și numărul pionului ce urmează să fie mutat, precum și numărul zarului și fereastra de joc pentru a putea modifica in cadrul interfeței grafice.

În metoda mutaPion ne folosim de o structură switch-case pentru a muta pionul în funcție de culoare, iar în cazul ambelor culori avem încă o structură de tip switch-case pentru a muta pionul în funcție de numărul acestuia. De asemenea ne mai folosim de două liste de coordonate din clasele Rosu și Albastru, liste ce reprezintă drumul pionilor pe tablă.

Inițial locațiile pionilor stocate in variabilele locatie_curentaAlbastru si locatie_curentaRosu au valorile -1, iar când pionii ies din casă, variabilele acestora o sa primească valoarea 0 pentru prima poziție, apoi cu cât avansăm în joc, cu atât o să crească valorile respective până la pozițiile 40, 41, 42 și 43, poziții ce reprezintă căsuțele de final ale jocului, unde jucătorul trebuie să aibă toți pionii pentru a câștiga.

```
if (locatie_curentaAlbastru[1] == -1 && numarzar == 6 )
{   locatie_curentaAlbastru[1] = 0;
    albastruIesit = true;

}
else if (locatie_curentaAlbastru[1] != -1)
{
    if (locatie_curentaAlbastru[1] + numarzar <= 43)
    {
        locatie_curentaAlbastru[1]+=numarzar;
    }
    else
    {
        int aux = numarzar;
        while (locatie_curentaAlbastru[1] < 43)
        {
            locatie_curentaAlbastru[1]++;
            aux--;
        }

        while (aux > 0)
        {
            locatie_curentaAlbastru[1]--;
            aux--;
        }

    }
}
if (locatie_curentaAlbastru[1] != -1)
{
    ferJoc.pAlbastru1.Location = albastru.Coordonate[locatie_curentaAlbastru[1]];
}
```

În porțiunea de cod alăturată avem un exemplu al unui case din această metodă. Primul if verifică daca pionul respectiv este în casă și zarul are cifra 6, în acest caz pionul iese din casă. Else if reprezintă situația unde pionul respectiv a ieșit deja din casă așa că acesta se mută în funcție de numărul zarului. Dacă suma dintre locația pionului și numărul zarului depășesc limitele tablei de joc, atunci folosim o variabilă ajutătoare pentru a face a pionul să facă mutările în exces cu spatele.
La final, în ultimul if reprezentat în această bucată de cod verificăm dacă s-a efectuat o mutare și ne ocupăm de aceasta și în interfața grafică, modificând coordonatele pionului.

Metoda mutaPion este apelată prin altă metodă, metoda Muta din clasa Joc, unde, pe lângă mutarea efectivă a pionului ne ocupăm de niște elemente ale logicii de joc:

- Verificăm dacă jucătorul are voie să mute (poate muta doar când este rândul său)
- Schimbăm rândul doar dacă numărul zarului este diferit de 6, astfel în cazul în care zarul este 6 putem să mai dăm cu zarul iar și să mutăm o piesă.

- Verificarea câștigătorului prin metoda verificareCastigator() din clasa Jucator, clasa pe care clasa Joc o moștenește.

Întreaga funcționalitate se rulează atunci când jucătorul dă click pe unul dintre pionii lui, în clasa FereastraJoc fiind apelată metoda Muta din clasa Joc, precum și trimiterea mesajelor în rețea pentru ca mișcarea să fie efectuată și în fereastra celuilalt jucător.

2.2.2 Pre-condiţii

Pentru a muta pionii pe tablă trebuie să efectuăm următoarele acțiuni:

- Să avem ambele aplicații deschise (client și server)
- În cazul în care jucăm pe un singur calculator trebuie să apăsam pe Start în aplicația server, iar dupa apariția mesajului ”Waiting for other player” apăsăm Start și în aplicația client care deschide apoi fereastra de joc, ulterior apăsam pe Ok in mesajul aparut pentru a deschide fereastra de joc și pentru server.
  În cazul în care jucăm pe 2 calculatoare urmăm aceeași pași cu excepția faptului că trebuie să introducem adresa IP afișată în aplicația server, în căsuța corespunzătoare IP ului.
- Odată ce am reușit să ajungem în fereastra de joc, trebuie să dăm click pe zar până când primim numărul 6, corespunzător cu rândul jucătorilor.
- Dăm click pe pionul pe care dorim să îl ducem pe tabla de joc
- Dăm click pe zar iar, și apoi dam click pe pion pentru a-l muta.

2.2.3 Post-condiţii

După rularea funcționalității putem observa cum pionul selectat își modifică poziția pe tablă corespunzător cu numărul zarului, moment în care ne dăm seama că funcționalitatea rulează corect.

3 Posibilitatea de a da cu zarul

3.1 Descriere

Asemenea mutării pieselor pe tablă, zarul este o funcționalitate nelipsită dintr-un joc de Nu te supăra frate, pionii mutându-se numai după numărul indicat de zar.

3.2 Fluxul de evenimente

3.2.1 Fluxul de bază

Această functionalitate pornește din clasa Zar prin metoda daCuZarul, unde variabila numar_zar primește un număr aleatoriu între 1 și 6, iar apoi modifică imaginea zarului din interfața grafică cu ajutorul unei structuri switch-case care schimba proprietatea imaginii în funcție de numărul zarului.

În clasa Zar avem de asemenea metodele getNumarZar() și setNumarZar() metode ce sunt esentiale pentru buna funcționare a programului, de exemplu, pentru a trimite și modifica în rețea valorile zarului.

```
if (joc.getRand() == "Albastru")
{
    numar_zar = zar.daCuZarul(this);

    if (joc.getRand() == "Rosu" && (!joc.getRosuIesit()) && numar_zar != 6)
        joc.setRand("Albastru");
    else if (joc.getRand() == "Albastru" && (!joc.getAlbastruIesit()) && numar_zar != 6)
        joc.setRand("Rosu");

    if (joc.getAlbastruIesit() == false && numar_zar < 6)
    {
        joc.setRand("Rosu");
        trimiteRand();
    }

    StreamWriter scriere = new StreamWriter(stream_server);
    scriere.AutoFlush = true; // enable automatic flushing
    scriere.WriteLine("z" + Convert.ToString(numar_zar));
    //se trimit datele printr-un string care incepe cu "z"
    //pentru ca la momentul primirii sa ne dam seama ca stringul prmit este o informatie despre zar
}
else
    MessageBox.Show("Nu este randul tau!");
```

În secțiunea de cod alăturată putem observa metoda DaCuZarul din clasa FereastraJoc, metodă care pe langă datul cu zarul, asigură respectarea unor reguli, precum schimbarea rândului atunci când nu avem un pion pe tabla și zarul are o valoare diferită de 6, trimiterea în rețea a valorii zarului și afișarea unui mesaj în cazul în care jucatorul încearcă să mute un pion deși nu e rândul său.

3.2.2 Pre-condiții

Pentru a rula această funcționalitate trebuie să urmăm următorii pași:

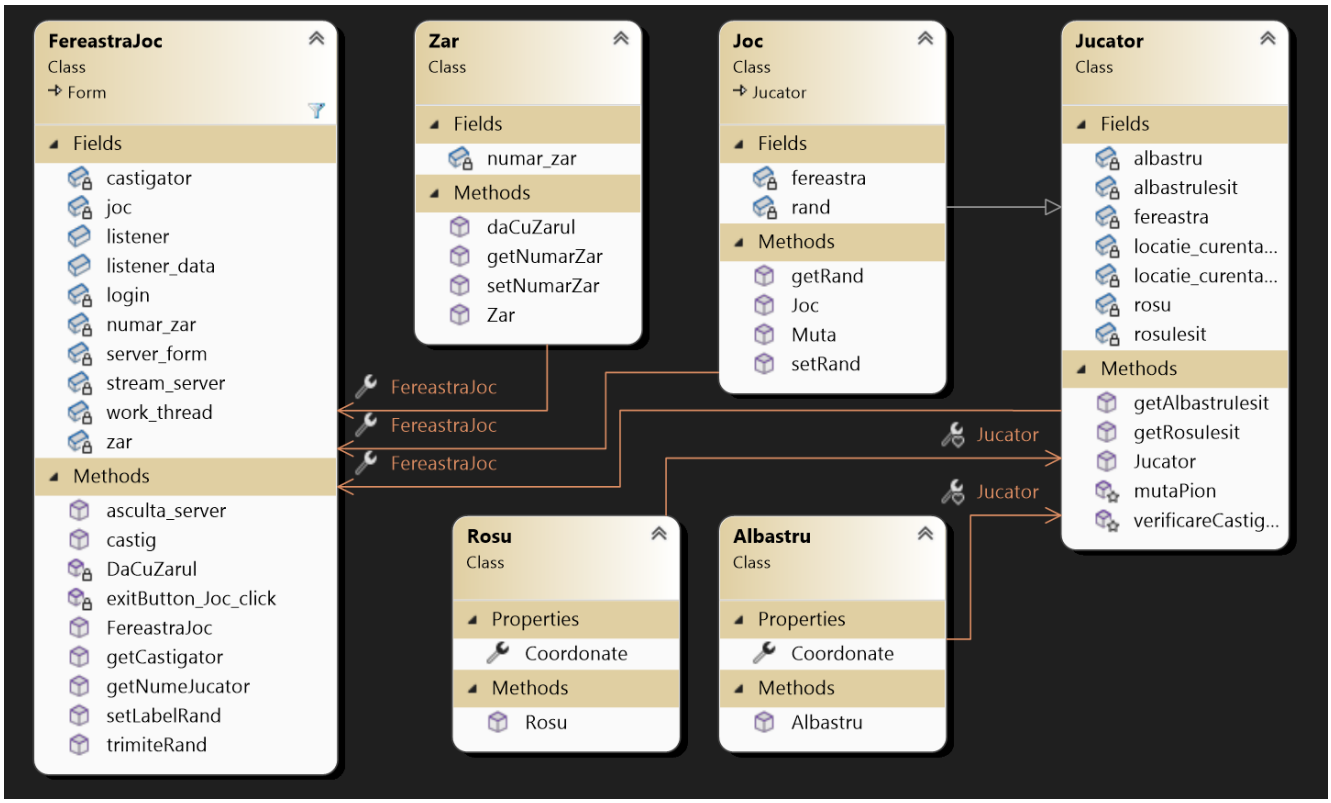
- Să avem ambele aplicații deschise (client și server)
- În cazul în care jucăm pe un singur calculator trebuie să apăsam pe Start în aplicația server, iar dupa apariția mesajului ”Waiting for other player” apăsăm Start și în aplicația client care deschide apoi fereastra de joc, ulterior apăsam pe Ok in mesajul aparut pentru a deschide fereastra de joc și pentru server.
  În cazul în care jucăm pe 2 calculatoare urmăm aceeași pași cu excepția faptului că trebuie să introducem adresa IP afișată în aplicația server, în căsuța corespunzătoare IP ului.
  Odată ce am reușit să ajungem în fereastra de joc, trebuie să dăm pur și simplu click pe zar.

3.2.3 Post-condiții

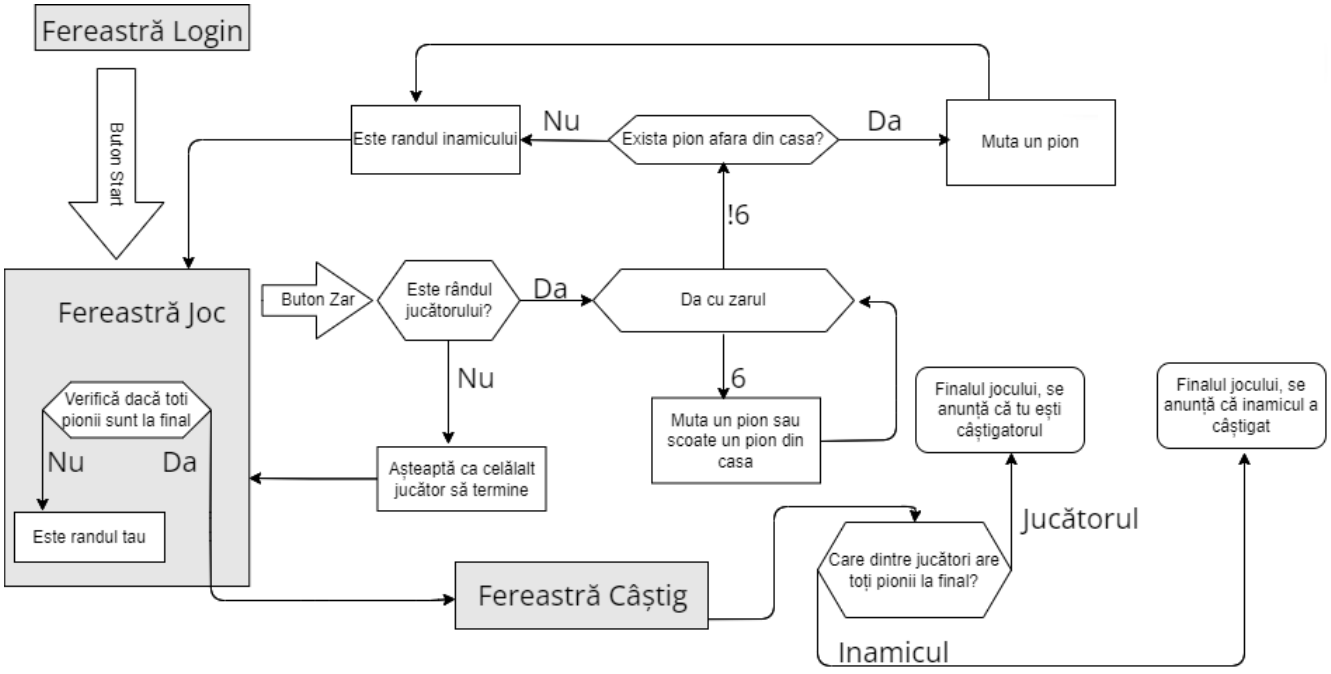
Putem observa ușor faptul că funcționalitatea functionează deoarece imaginea zarului se modifică după ce dăm click(cu exceptia cazului în care numărul primit este același cu cel afișat inițial, caz în care observăm schimbarea doar când primim un număr diferit) și atunci când mutăm pionii putem observa că aceștia se deplasează cu un număr egal de casute cu numărul afișat pe zar.

4 Implementare

4.1 Diagrama de clase



4.2 Descriere detaliată



5 Bibliografie

- GitHub Copilot - github.com/features/copilot
- JetBrains ReSharper - jetbrains.com/resharper
- GitHub - github.com
- OpenAI ChatGPT - chat.openai.com
- VisualParadigm Online - online.visual-paradigm.com
- Visual Studio Forms App – Create a Visual Studio Forms App with C#