Proiect 1

Joc

Să se implementeze o aplicație client-server în Java pentru un joc cu mai mulți jucători. Fiecare jucător va fi reprezentat de către un client software, aplicația server fiind cea care coordonează jocul.

Regulile jocului sunt următoarele:

a) numărul de jucători ce participă la o partidă este de minim doi şi de maxim cinci. Fiecărui jucător îi este atribuit un simbol (poate fi o culoare, o literă, o iconiță sau orice altceva ce permite identificarea unui jucător pe tabla de joc). De exemplu, fiecare jucător ar putea primi o literă colorată '`R`' – *red*, '`G`' – *green*, '`B`' – *blue*, '`Y`' – *yellow*, '' - .

b) Atunci când îî vine rândul, fiecare jucător va amplasa câte o piesă pe tabla de joc (la fel ca la jocul de Go[1]).

c) Tabla de joc are dimensiunile: 6 linii şi 10 coloane. La început, serverul va plasa în mod aleator câte o piesă pentru fiecare jucător (corespunzător pentru fiecare culoare). Piesa corespunzătoare unui jucător este plasată de către server pe tablă numai după ce acesta se înscrie să participe la partidă.

d) Jucătorii efectuează mutări, fiecare atunci când îi vine rândul. Ordinea de efectuare a mutărilor este aceeaşi cu ordinea de înscriere pentru participare la partidă (ordinea de conectare la server). Putem presupune că primul jucător a fost cel cu simbolul '`R`', urmat de cel cu simbolul '`G`' şi apoi jucătorul '`B`'.

e) La fiecare mutare, un jucător va amplasa o piesă pe tablă. El poate amplasa o piesă numai într-o celulă liberă şi care are cel puțin o celulă vecină în care există o piesă de acelaşi tip (de aceeaşi *culoare*).

f) Fiecare jucător primeşte *trei jokeri* la începutul jocului. Un jucător poate utiliza oricare dintre aceştia atunci când face o mutare. Odată ce un joker este utilizat, el nu mai poate fi utilizat a doua oară de aceaşi persoană. Un joker îi conferă avantajul celui care îl utilizează de a nu respecta regula generală a jocului:

    a. Jokerul "*mutare dublă*" permite jucătorului să amplaseze două piese la o singură mutare – este vorba de două mutări consecutive.

    b. Jokerul "*înlocuire*" permite jucătorului să amplaseze o piesă într-o celulă chiar dacă aceasta nu este goală. Celula unde se realizează mutarea trebuie însă să aibă cel puțin o celulă vecină în care să existe o piesă de acelaşi tip cu cea plasată.

[1] https://en.wikipedia.org/wiki/Go_(game)

c. Jokerul "*libertate*" permite jucătorului să amplaseze o piesă într-o celulă liberă.

g) Un jucător este *blocat* dacă el nu poate face nicio mutare, atunci când îi vine rândul, chiar dacă ar folosi şi un joker dintre cele disponibile la acel moment. Dacă un jucător este *blocat*, atunci el îşi *pierde mutarea* (se trece la următorul).

h) *Partida se termină* atunci când toți jucătorii sunt *blocați* (când niciunul dintre jucători nu mai poate efectua o mutare).

i) Câştigătorul partidei este jucătorul care are *cel mai mare număr de piese* amplasate pe tablă la final. Dacă există mai mulți jucători cu acelaşi număr de piese, câştigă cel care mutat ultimul dintre aceştia. De exemplu, dacă la final avem situația următoare '`R`': 21, '`G`': 18, '`B`': 21, câştigă jucătorul '`B`' deoarece el este cel care a mutat mai târziu decât '`R`'.

A) Jocul şi interfața pentru utilizator

Să se realizeze o aplicație client-server.

- Serverul se va implementa utilizând socketi sau sub forma unui serviciu web în Jersey[2].
- Clientul poate fi o aplicație Java cu o interfață grafică sau o interfață text sau o pagină Web HTML/JavaScript.
- Serverul trebuie să fie cel care verifică corectitudinea şi ordinea mutărilor.
- Se recomandă să se implementeze teste unitare ce verifică buna funcționare a serverului.

B) Documentația

Va fi realizată sub forma unui raport Word şi PDF şi va include:

- o descriere detaliată a modului de pornire a serverului şi a clientului din linia de comandă sau din browser.

- o descriere a arhitecturii aplicației. Dacă s-au utilizat servicii Web se va descrie API-ul. Dacă s-au folosit socketi, se va descrie protocolul utilizat. Se vor prezenta, pe scurt, testele unitare. Se va explica cum se comportă soluția propusă în cazul accesului concurent din partea mai multor clienți.

- se vor prezenta diagrame use-case, diagrame de clase, diagrame de secvență, diagrame de colaborare, diagrame de stare etc., corespunzătoare aplicației software realizată.

- se vor prezenta. dacă este cazul, şabloanele de proiectare folosite.

- o descriere a interfeței şi a deciziilor luate pentru a o face mai prietenoasă cu utilizatorul.

- concluzii: cum s-a desfăşurat proiectul; care părți au fost mai simple, care au fost mai dificile; există unele decizii arhitecturale pe care vrem să le prezentăm în mod deosebit; care sunt părțile neimplementate, parțial implementate sau total implementate; cum ne-am

[2] https://eclipse-ee4j.github.io/jersey/

descurcat cu gestiunea timpului în ceea ce priveşte realizarea proiectului; sunt aspecte pe care le-am realiza într-un mod diferit la următoarea implementare.

Observații

Arhiva proiectului va fi sub forma unui Project Eclipse[3] şi va include codul sursă, documentația, librării utilizate şi alte fişiere necesare rulării aplicației pe un alt calculator.

Se va încărca pe Google Classroom o arhivă cu numele `proiect-1-AAA.zip` unde `AAA` reprezintă numele şi prenumele studentului. Arhiva va conține codul sursă, fişiere de configurare, instrucțiuni pentru lansarea în execuție, documentația (fişiere Word şi pdf) etc.

[3] https://www.eclipse.org/