Battleship game specifikacija

Naredna poglavlja sadrže informacije o dijagramima komponenti, slučajevima upotrebe, sekvenci, UML dijagramu glavnim glasa, kao i opis slučaja upotrebe.

Projekat, Unit testovi, kao i sva projekta dokumentacija se nalaze na:

https://github.com/MATF-RS20/RS002-battleship-game

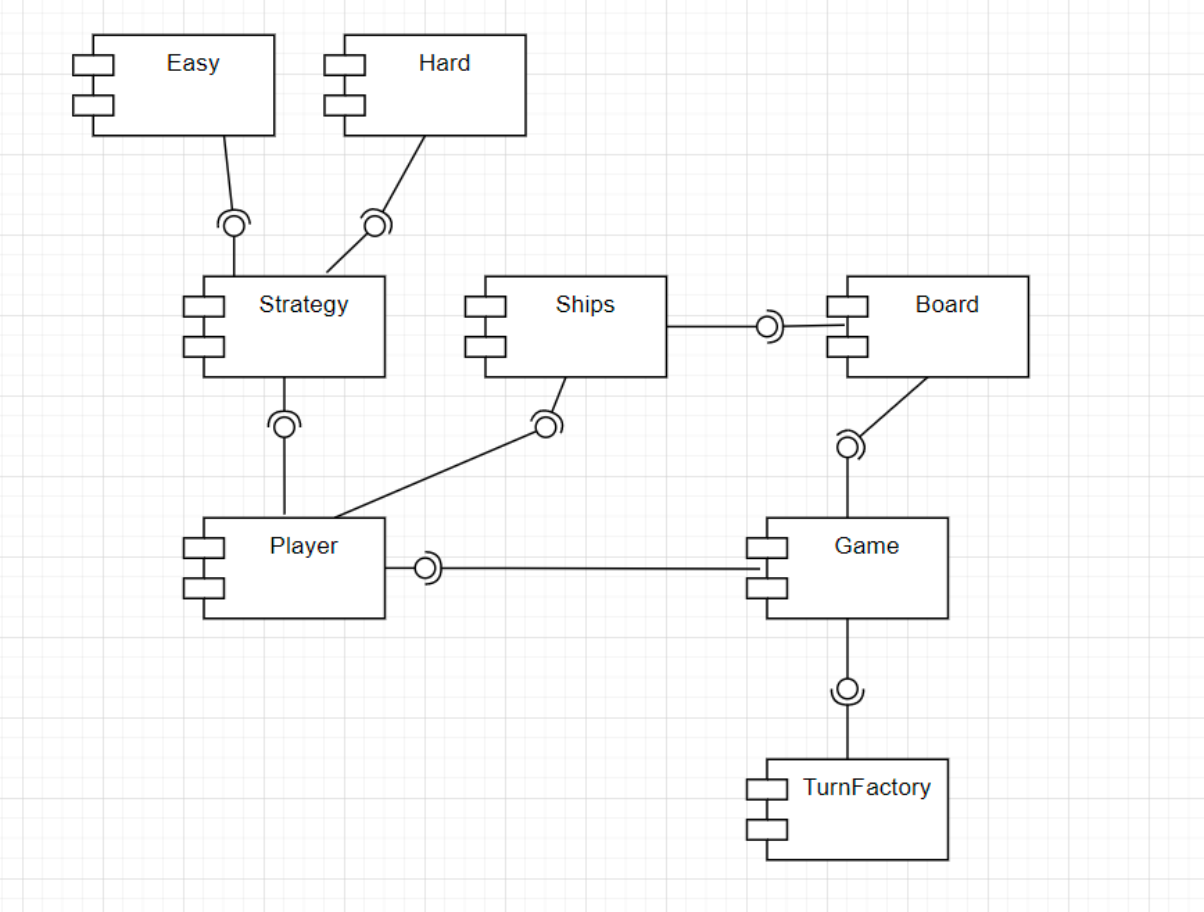
1. Uputstvo za pokretanje projekta

Izvorni kod aplikacije je moguće preuzeti sa linka iznad. Kao preduslov za bilodvanje, potrebno je instalirati Qt5 framework zajedno sa njegovim razvojnim okruženjem – Qt Creator.

Nakon učitavanja .pro fajla u QtCreator, potrebno je konfigurisati projekat tako da podržava x86 ili x64 platformu. Nakon toga, potrebno je izbildovati projekat I pokrenuti ga uz pomoć Run komande.

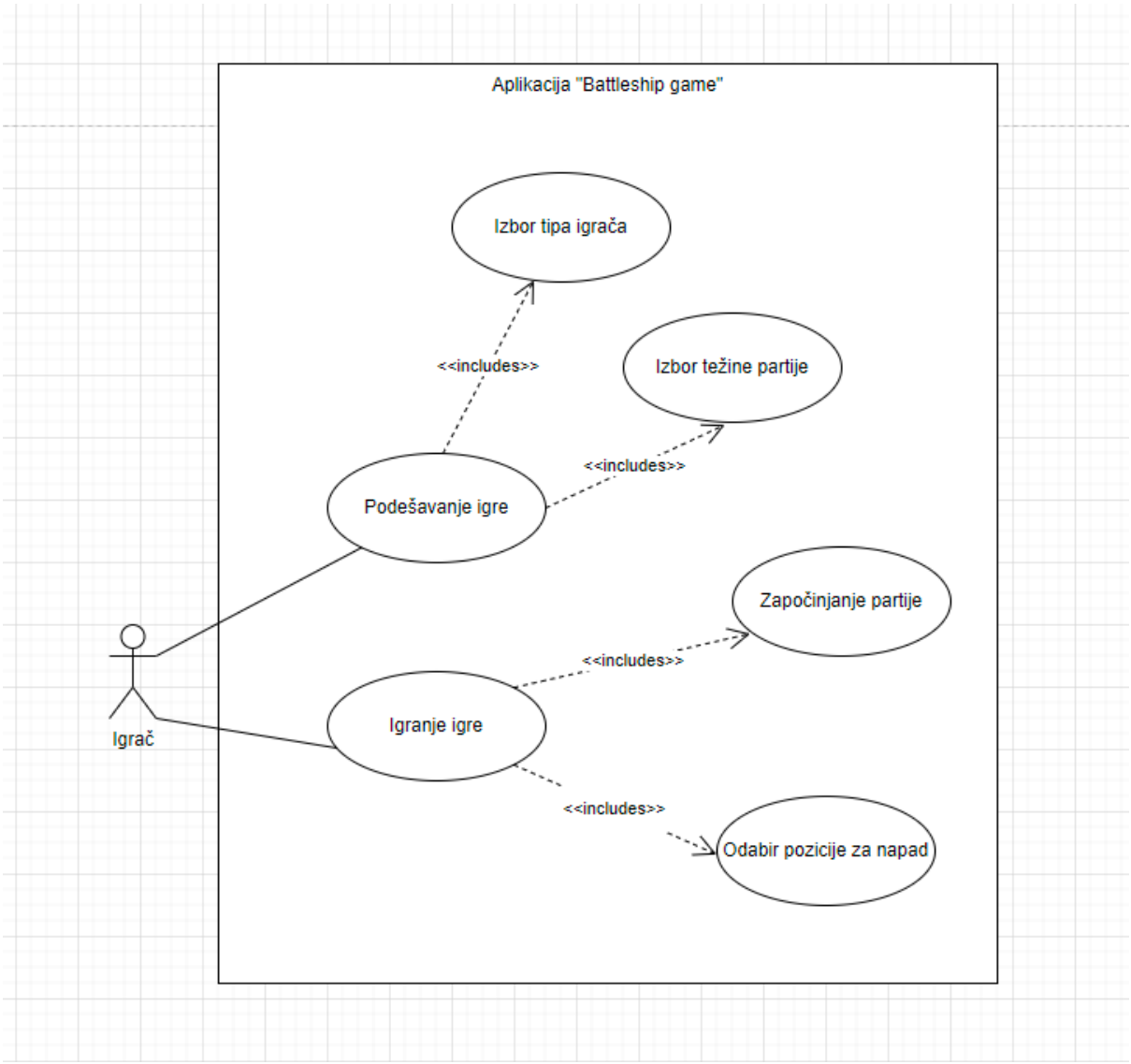
2. Dijagram komponenti

Osnovne komponente aplikacije su Igrač, Brodovi, Tabla, Strategija koju igrači koriste, kao i sama Igra. Međusobne povezanosti su prikazane na dijagramu komponenti:



3. Dijagram slučajeva upotrebe

Prikazani slučajevi upotrebe uključuju podešavanje i igranje igre. Osnovni smisao podešavanja igre je podešavanje tipa igrača. Tip igrača može biti računar ili čovek. U slučaju da je igrač računar, on može biti u “lakšem” i “težem” režimu/nivou.



4. Opis slučaja upotrebe

Naziv: "Odabir pozicije za napad"

Kratak opis: Igrač odabira jednu poziciju sa table za koju sumnja da se iza nje nalazi brod. U slučaju da se brod zaista nalazi iza te pozicije, igrač ima pravo da igra ponovo, sve dok ne promaši. Nakon promašaja, drugi igrač je na potezu. Igrač može biti čovek ili računar. U slučaju da je to računar, implementirani su algoritmi kojima se dolazi do pozicije za napad (Easy i Hard mode).

Akteri:

- Igrač (Computer Player)

Preduslovi: Igra je pokrenuta i bar jedan igrač je računar. Računar je na potezu.

Osnovni tok:

1. Računar je na potezu da izvrši napad.

2. Aplikacija prikazuje tablu protivničkog igrača.

3. Aplikacija prelazi u režim napada igrača - računara.

4. Aplikacija kreira novi potez.

5. Aplikacija kreira napad.

5.1. U slučaju da je igrač (računar) podešen na lak režim (easy mode), algoritam generiše pozicije za napad nasumično. Računar ne pamti prethodne napade i ne uzima ih u obzir prilikom izbora narednog.

5.2. U slučaju da je igrač (računar) podešen na težak režim (hard mode), algoritam generiše pozicije za napad uzimajući u obzir prethodno izvšene napade.

6. Aplikacija izvršava napad na protivničku tablu.

7. Aplikacija proverava status pozicije na koju je izvršen napad.

7.1. U slučaju da protivnički brod nije pogođen, igrač gubi pravo da ponovo gađa i protivnički igrač je na potezu.

7.2. U slučaju da je protivnički brod pogođen, igrač je ponovo na potezu.

7.2.1. U slučaju da nisu svi protivnički brodovi potopljeni, igrač bira novu poziciju za napad. Prelazi se na korak 1.

7.2.2. U slučaju da su svi protivnički brodovi potopljeni, igrač je pobedio.

Alternativni tokovi:

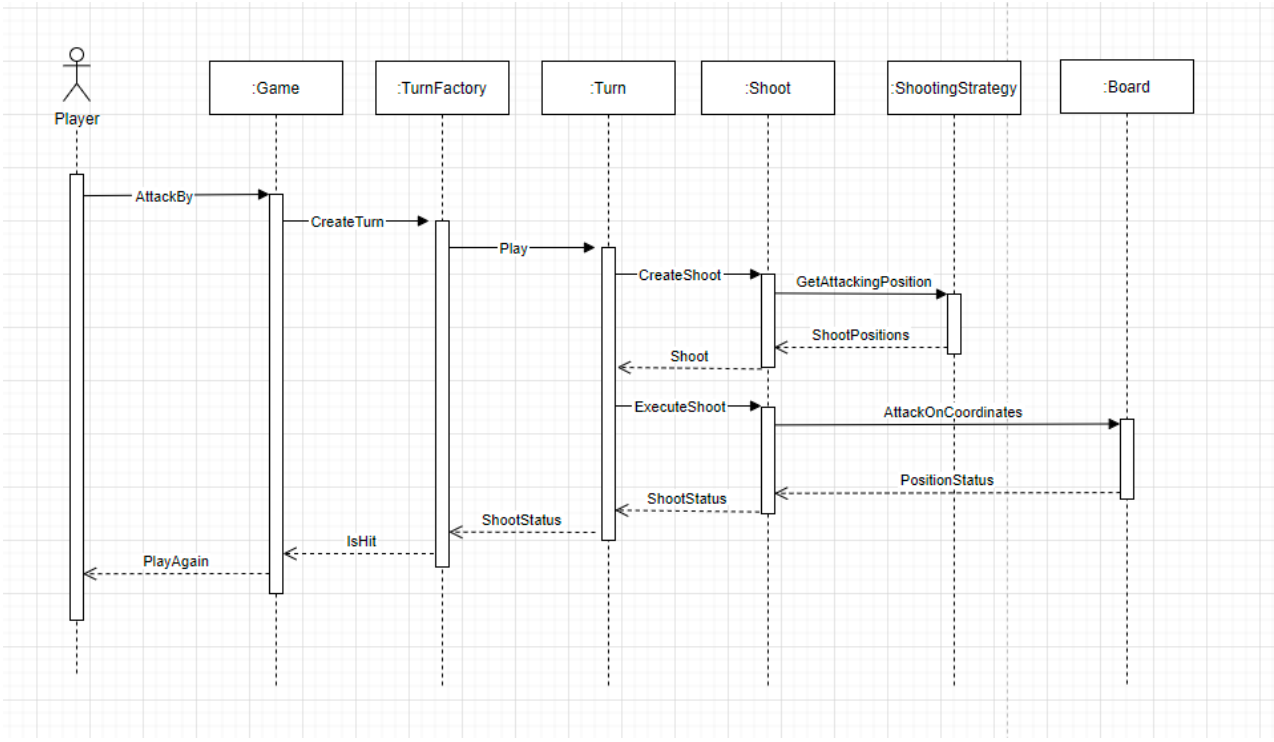
Ime: Neočekivani izlaz iz aplikacije.

Opis: Ukoliko u bilo kojem koraku korisnik isključi aplikaciju, sve eventualno zapamćene informacije o trenutnoj partiji igre se poništavaju i aplikacija završava rad. Slučaj upotrebe se završava.

Dodatne informacije: Tokom trajanja partije, aplikacija pamti informacije za svakog igrača o pozicijama na koje su izvršeni napadi. U slučaju težeg režima (hard mode), te informacije služe kao ulazni podaci prilikom odabira naredne pozicije na koju će biti izvršen napad.

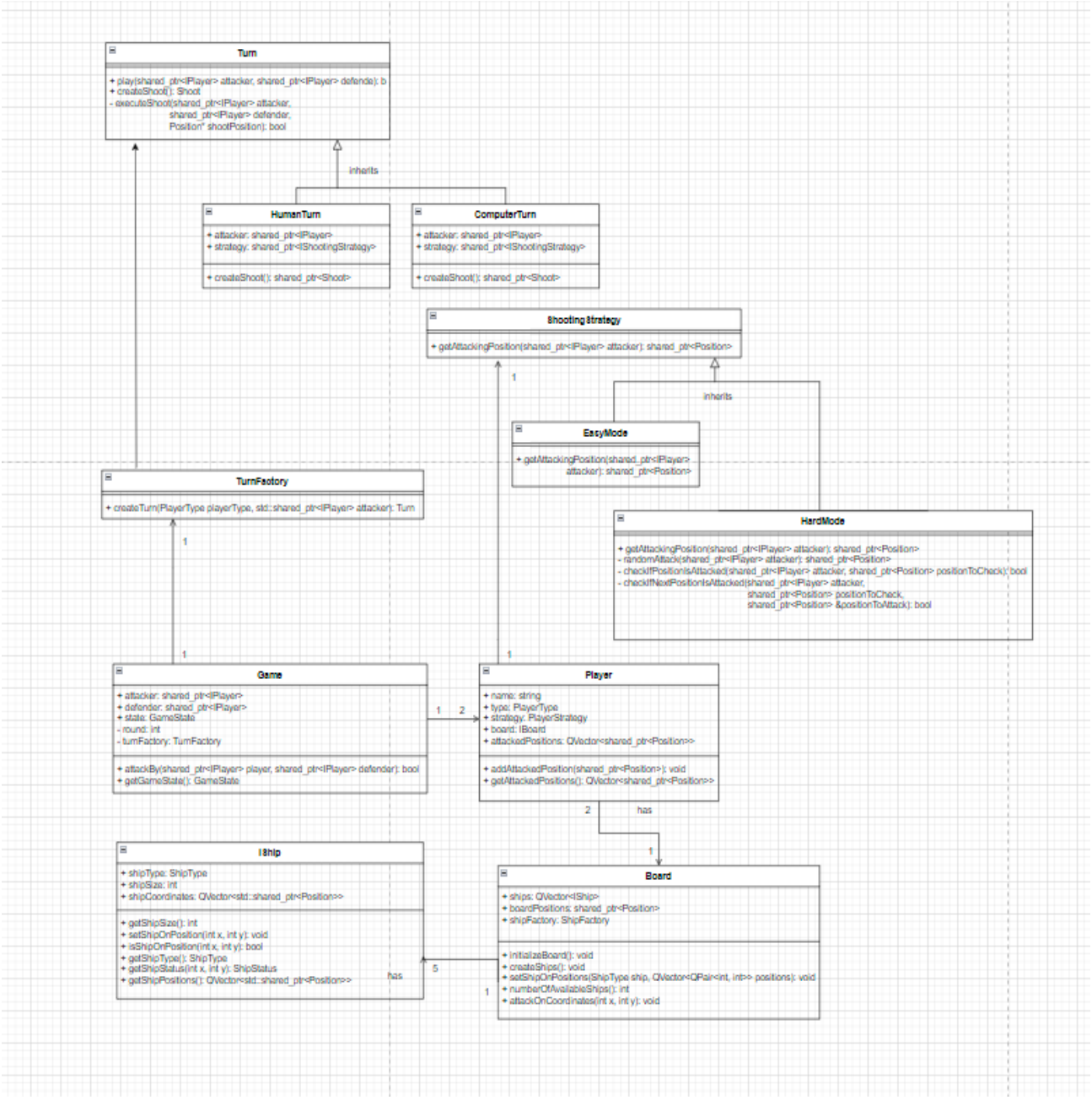
5. Dijagram sekvenci

Prikazani dijagram sekvenci opisuje zavisnosti između klasa I tok akcija koji se preduzima u slučaju upotrebe opisanom u poglavlju 3.



6. Dijagram glavnih klasa

U nastavku je prikazan UML dijagram glavnih klasa koji prikazuje detaljniju specifikciju samih klasa – njihovih podataka članova I važnijih metoda, kao I zavisnosti između različitih klasa.



Ovako prikazan UML dijagram klasa, zajedno sa svim prethodnim dijagramima, bi trebalo da predstavlja polaznu tačku za implementaciju same aplikacije.