Imperit

Strategická tahová hra

Imperit

Turn-based strategy

Závěrečná maturitní práce, rok 2021
Richard Blažek
Gymnázium Brno, třída Kapitána Jaroše 14

Prohlášení

Prohlašuji, že jsem svou závěrečnou maturitní práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze závěrečné maturitní práce jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Brně dne 18. března 2021 .....

Poděkování

Tímto bych chtěl poděkovat Mgr. Marku Blahovi za odborné vedení práce.

Anotace

Práce se zabývá vytvořením internetové počítačové hry zvané Imperit pomocí Blazor WebAssembly z frameworku .NET. Hra je koncipovaná jako tahová strategie, jejímž tématem je dobývání území na herním plánu, s právě jedním vítězem a omezeným náhodným prvkem.

Klíčová slova

počítačová hra; tahová strategie; válečná strategie; územní expanze; hra s nulovým součtem; dotnet; blazor; webassembly

Annotation

The thesis is concerned about creation of online browser-based game called Imperit using Blazor WebAssembly from the .NET framework. The game is designed as a turn-based strategy consisting of conquering the territory on the game map, having exactly one winner and a limited element of chance.

Keywords

computer game; turn-based strategy; war strategy; territorial expansion; zero-sum game; dotnet; blazor; webassembly

Obsah

1 Úvod

Rozhodl jsem se vytvořit hru o dobývání území navrženou tak, aby byla konečná a vítěz byl jednoznačně určen. Vítězství by však nemělo na konci hry nastat neočekávaně, již v průběhu hry bude patrný vývoj, z něhož vyplyne, který hráč má k výhře nejblíže, ale tento vývoj budou moci ostatní hráči zvrátit. Hra by také měla obsahovat náhodný prvek, ovšem pravděpodobnost náhodných jevů by měla být známá, aby byli hráči nuceni s rizikem počítat.

Pro vytvoření hry jsem zvolil formát tahové strategie, protože strategie v reálném čase buď vyvíjí tlak na hráče z důvodu nedostatku času, nebo ve snaze vyhnout se tomuto problému vede ke zdlouhavému čekání na dokončení některých akcí [1]. V obou případech realtimová strategie vede k orientaci na postřeh a trpělivost spíše než na vymýšlení strategie[2]. Kromě toho „připoutává“ hráče k jejich obrazovkám, což sice může být mnohdy žádoucí, ale můj záměr takový nebyl.

2 Popis hry

Hru může hrát 2–16 hráčů. Hra se odehrává na mapě, která se skládá ze 126 provincií, jejichž názvy a tvary přibližně odpovídají evropským státům, krajům, mořím a pohořím. Cílem hry je získat alespoň tři z cílových zemí, kterými jsou Anhaltsko, Katalánsko, Kosovo a Královec. K tomu je sice nutné ovládnout podstatnou část mapy, ale při vhodně zvolené strategii může hráč vyhrát, i když bude mít menší území než ostatní hráči.

2.1 Registrace

Při registraci si hráč zvolí jméno a heslo a zobrazí se mu mapa, na níž si kliknutím vybere zemi, kde bude začínat. Na výběr má pouze některé ze zemí. Název zemí, v nich nelze začínat, je napsán šedou barvou. Po registraci hráč musí počkat, než hra začne. Po registraci druhého hráče čeká hra na registraci dalších hráčů čtyři minuty a potom se automaticky spustí.

2.2 Průběh hry

Hráč začíná s určitým množstvím peněz a s vojáky ve své počáteční provincii. Tahy hráčů se střídají v tom pořadí, v němž se tito hráči zaregistrovali. Během tahu může hráč provádět libovolné množství akcí bez časového omezení, dokud se sám nerozhodne svůj tah ukončit. Může verbovat vojáky, kupovat a dobývat země. Verbování a dobývání se však dokončí až po ukončení tahu.

2.3 Provincie

Jsou tři typy provincií: země, moře a pohoří. Země a moře můžou patřit nějakému hráči, který v nich může mít svou armádu. Pohoří nikomu nepatří a armáda v nich být nemůže. Hráč dobude provincii, pokud do ní pošle armádu, jejíž útočná síla je vyšší než obranná síla tamních

jednotek, a ovládá ji, dokud mu ji stejným způsobem nedobude jiný hráč, nebo dokud se tato provincie sama neodtrhne.

Země se může hráči odtrhnout vždy po jeho tahu. Pravděpodobnost odtržení země bez armády je 10 % a s růstem obranné síly armády klesá tak, že když obranná síla armády v zemi je alespoň tak velká jako na začátku hry, země se již nemůže odtrhnout.

2.4 Peníze

Hráč získává po svém tahu daňový výnos ze svých zemí a tento výnos se u různých zemí liší. Jestli chce hráč získat výnos ze všech svých zemí, musí jeho provincie tvořit souvislé území. Pokud je území hráče rozdělené na více částí, mezi nimiž jsou provincie, které jsou neobydlené nebo jsou pod kontrolou jiného hráče, získá hráč pouze výnos z části s nejvyšším výnosem. Existuje tedy značná motivace udržovat své území souvislé.

Za peníze je možné kupovat další země nebo vojenské jednotky, s nimiž lze dobývat provincie a bránit provincie už dobyté. Cena země je součtem ceny jejích vojenských jednotek, dvojnásobku jejího výnosu a obranné síly jejích vojenských jednotek. Pokud hráč dluží nějaké peníze, jsou mu po získání daňového výnosu zabaveny peníze ke splacení dluhu. Pokud všechny jeho peníze ke splacení dluhu nestačí, je zbývající dluh navýšen o úrok 25 % a zůstává hráči do dalšího kola. Dluh nesmí přesáhnout 400, pokud k tomu dojde, provede se exekuce, hráč ztratí několik svých zemí a jeho dluh bude snížen o cenu těchto zemí.

2.5 Vojenské jednotky

V tabulce jsou uvedeny typy vojenských jednotek, které se ve hře dají koupit. Je zde uvedena jejich cena, síla v boji, vzdálenost, na jakou se mohou samy přesunout a v jakém prostředí. Pokud se jednotka dokáže přesunout a navíc má nějakou nosnost, může přepravit jiné jednotky, které to nedokáží. Celková nosnost jednotek, které přepravu provádějí, musí být stejná nebo vyšší než je celková hmotnost přepravovaných jednotek.

Název	Cena	Síla	Vzdálenost	Nosnost	Hmotnost	Prostředí
Pěšák	1	1	1	–	1	země
Loď	200		1	200	500	země s přístavem/moře
Slon	6	8/2	2	–	10	země/pohoří
Fénická loď	50	0	2	400	500	země s přístavem/moře
Veslice	130	130	1	120	500	země s přístavem/moře

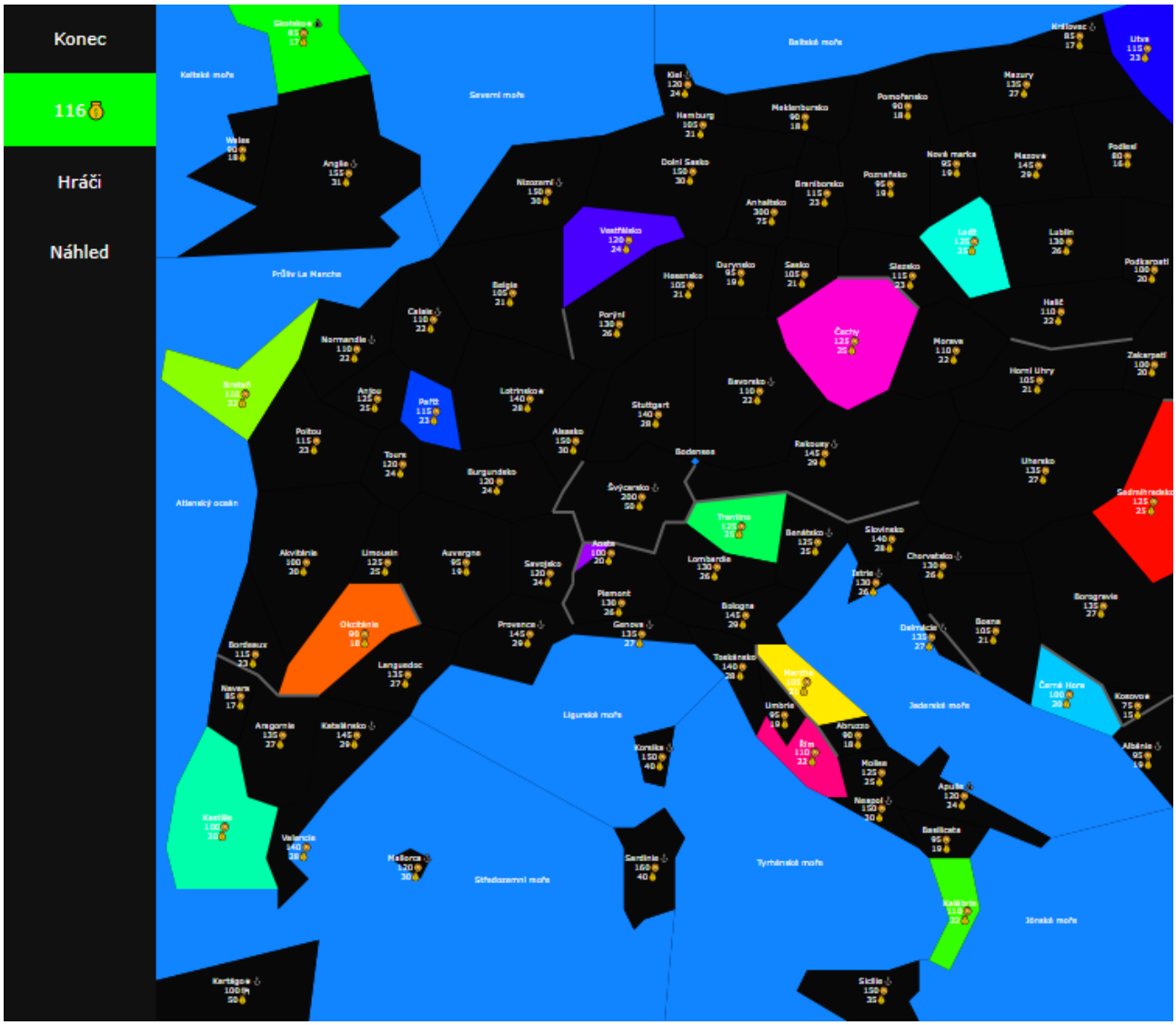
Slon může pohoří přecházet, ale nemůže v nich zůstat, je u něj uvedena před lomítkem síla v útoku z lomítkem v obraně. Slona a fénickou loď lze koupit jen v Kartágu, veslici (celým názvem „germánská bojová veslice“) jen v zemích s přístavem u Bodensee.

2.6 Ovládání

Na okraji obrazovky (levý okraj na širokých obrazovkách, horní na úzkých) je navigační menu, v němž může hráč přepínat mezi stránkami. Při přihlášení se jako výchozí stránka zobrazuje mapa provincií.

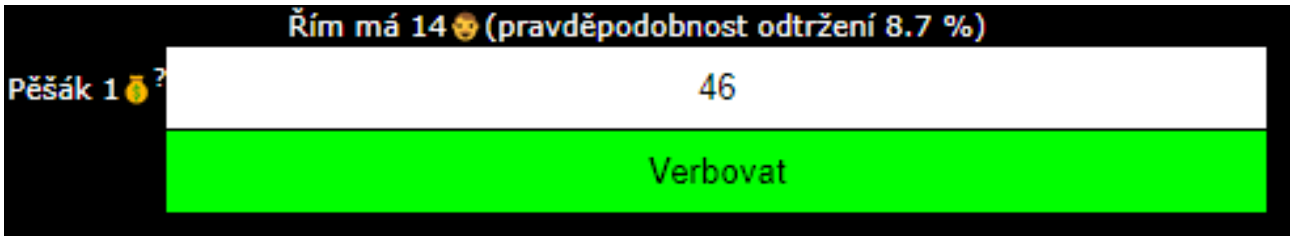
2.6.1 Mapa

Na této stránce hráč vidí mapu všech provincií. Pohoří se zobrazují jen jako šedá čára mezi zeměmi. Pokud jim žádný z hráčů nevládne, jsou moře tyrkysová a země černošedé; pokud v nich však někdo vládne, mají barvu onoho hráče. Dále se u moří a zemí zobrazuje jejich název a počet vojenských jednotek, u zemí se navíc zobrazuje jejich výnos za jedno kolo. Země, které mají přístav, mají u svého názvu kotvu, a země, které hráč potřebuje k vítězství, u svého názvu mají hvězdičku.



Klikne-li hráč dvakrát na svou provincii, může v ní verbovat jednotky; klikne-li na ni jednou a poté klikne na druhou provincii, může přesunout jednotky z první provincie do druhé; klikne-li na cizí zemi, jež sousedí s jeho zemí, může tuto zemi koupit.

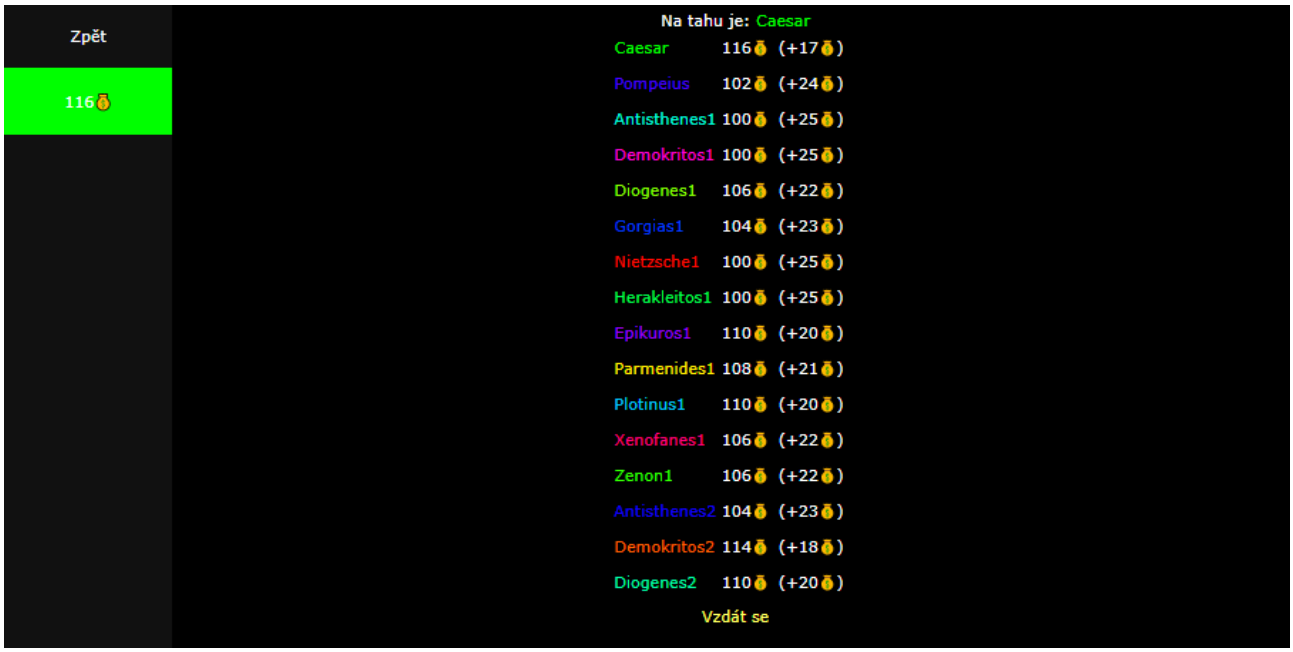
Při verbování a přesunu se ukáže nabídka, v níž může hráč vybrat, kolik jednotek kterého druhu chce verbovat (resp. přesunout). Pokud se hráč rozhodne verbovat více jednotek, než si může za své peníze koupit, nebo si koupit zemi, která stojí více peněz, než kolik daný hráč má, umožní mu hra si chybějící částku půjčit. Pokud se země může odtrhnout, zobrazí se v této nabídce pravděpodobnost, že k tomu dojde.



Navigační menu na této stránce má čtyři položky. První z nich je konec tahu, nebo odhlášení, podle toho, zda je přihlášený hráč na tahu. Druhá položka pouze informuje o tom, kolik peněz má přihlášený hráč. Třetí položka přepíná na stránku *Hráči* a poslední možnost ukazuje náhled mapy po provedení akcí hráče na tahu.

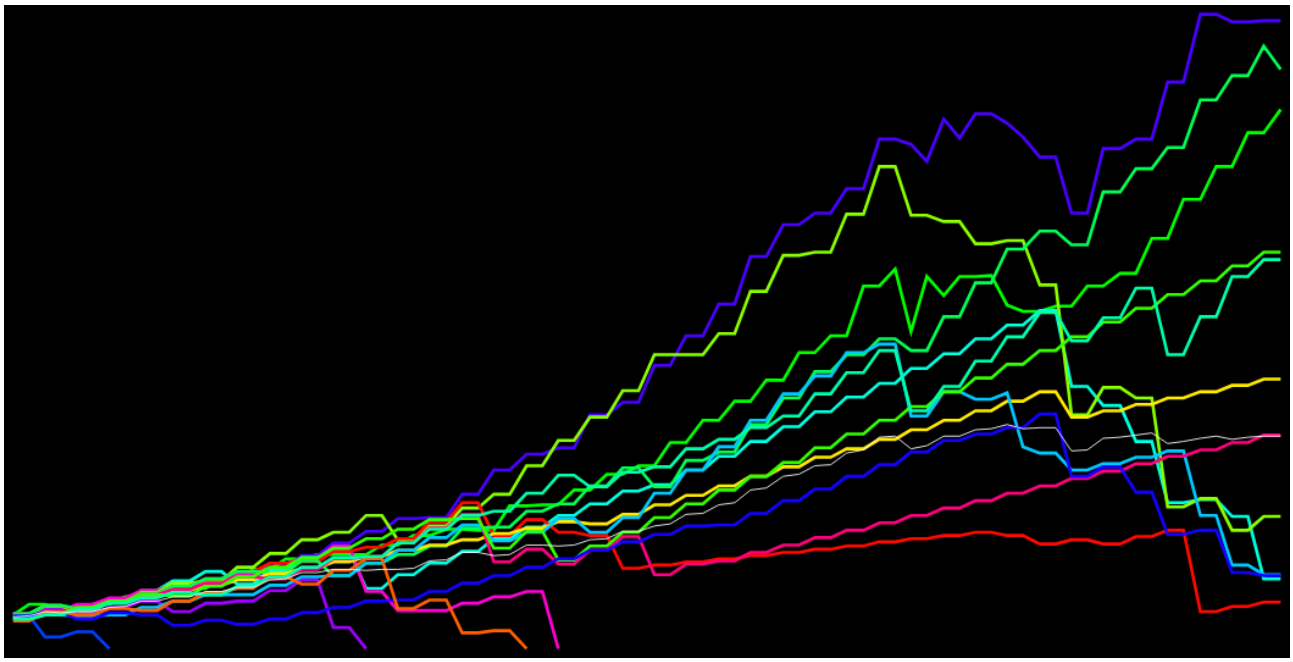
2.6.2 Hráči

Na této stránce se zobrazuje seznam všech hráčů, jejich peníze, příjmy, případě výše jejich dluhů. Klikne-li hráč na jméno jiného hráče, může mu věnovat peníze. Zobrazí se nabídka a v ní lze vybrat, kolik peněz darovat. Pod seznamem hráčů je možnost vzdát se, kliknutím na ni hráč zemře a všechny jeho provincie se odtrhnou.



Po proběhnutí prvních několika tahů se pod možností vzdát se zobrazí tři grafy. První z nich popisuje vývoj celkových sil hráčů (počítají se jako součet ceny všech vojáků, peněz, pětinásobku příjmů a stonásobku počtu cílových zemí, které daný hráč vlastní). Na druhém grafu se zobrazuje poměrná změna této síly. Na třetím se zobrazuje poměr vojenských sil hráčů (počítají se jako součet ceny všech vojáků a peněz) vůči sobě.

Na obrázku níže je ukázka prvního grafu ze hry, kterou vyhrál hráč, jenž má podle grafu až třetí největší sílu a jehož území zdaleka nebylo největší. Je tedy vidět, že při běžné hře k tomu může dojít.



3 Použité technologie

3.1 C#

Celý program je napsaný v jazyce C#, jelikož tento jazyk může běžet na straně serveru (viz ASP.NET Core) i klienta (viz Blazor WebAssembly) a obě části mohou využívat společné knihovny a pracovat s týmiž datovými typy. Navíc tento jazyk umožňuje využívat knihoven z frameworku .NET a na rozdíl od jazyků PHP a JavaScript, které se často pro vývoj webových aplikací používají, je staticky typovaný a kompilovaný do bytekódu, což do určité míry kontroluje správnost programu.

3.2 ASP.NET Core

Framework ASP.NET Core je využíván na straně serveru ke zpracovávání HTTP dotazů a odesílání odpovědí.

3.3 Blazor WebAssembly

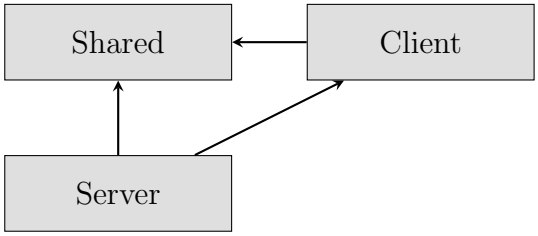
Blazor WebAssembly je framework, který umožňuje vyvíjet v jazyce C# aplikace spustitelné v prohlížeči. Kód v jazyce C# se přeloží do binárního formátu WebAssembly a prohlížeč pomocí krátkého kódu v JavaScriptu výsledný soubor stáhne a spustí. Spuštěný program následně reaguje na akce uživatele, komunikuje se serverem pomocí HTTP a podle potřeby překresluje zobrazovanou webovou stránku.

4 Program

Program se skládá ze tří projektů:

- *Shared* – Obsahuje třídy, které druhé dva projekty využívají, nachází se v něm většina herní logiky a odpovídá větší části modelu v architektuře MVC[3]
- *Client* – Odpovídá pohledu v architektuře MVC a obsahuje soubory v HTML, CSS a kód v C#, překládaný do WebAssembly a spouštěný v prohlížeči klienta
- *Server* – Implementuje webové API, jež odesílá odpovědi na dotazy klienta, a zahrnuje kód, který je potřebný k funkčnosti API, tedy práci s databází a s třídami z projektu *Shared*; v architektuře MVC odpovídá kontroleru a části modelu

Graf závislostí mezi projekty:



4.1 Shared

Projekt *Shared* obsahuje složku *Conversion*, v níž jsou třídy zajišťující načítání některých typů v ostatních složkách z formátu JSON nebo naopak zápis do tohoto formátu. Ve složkách *Commands*, *Config* a *Data* jsou datové typy, které odpovídají různým jevům ve hře. Jedná se o neměnné datové typy a z pohledu jazyka C# jsou to záznamy (*records*), nikoliv třídy. Dále je však budu označovat jako třídy, protože v paradigmatu objektově orientovaného programování se skutečně o třídy jedná, neboť jejich instance obsahují data i funkce, které s těmito daty pracují.[4] Neměnné typy umožňují psát funkce a metody, které jsou referenčně transparentní (tj. nemají vedlejší účinky), a tím usnadňují přemýšlení o programu a snižují riziko chyb. [5][6]

4.1.1 Commands

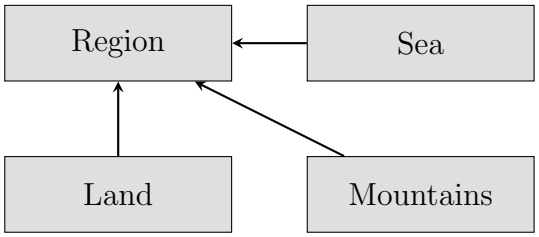
Ve složce *Commands* se nachází rozhraní *ICommand* a třídy, jež ho implementují. Každá z nich odpovídá nějakému příkazu, který může hráč provést: nákup provincie, darování peněz, verbování vojáků, přesun vojáků, kapitulace a konec tahu. Rozhraní má dvě metody; obě přebírají jako paramtetry seznam hráčů a provincií, nastavení hry a hráče, jenž příkaz provedl. Jedna z nich ověřuje, zda je příkaz možné provést a druhá jej provede a vrátí změněný seznam hráčů a provincií (aniž by měnila ten původní). Kód rozhraní vypadá takto:

```
public interface ICommand
{
    bool Allowed(Player actor, IReadOnlyList<Player> players, Provinces provinces, Settings settings);
    (IEnumerable<Player>, IEnumerable<Province>, Game) Perform(Player actor, IReadOnlyList<Player> players,
                            Provinces provinces, Settings settings, Game game);
}
```

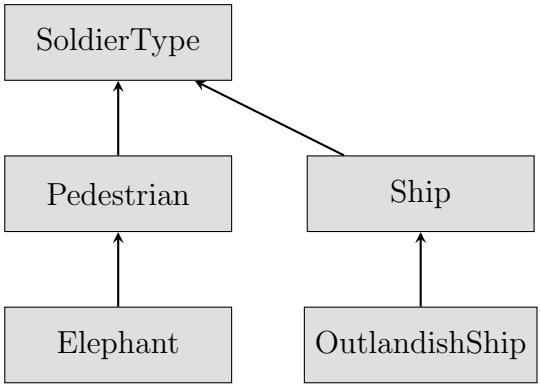
4.1.2 Config

Ve složce *Config* se nachází třída *Settings*, jež obsahuje data načtená při spuštění programu z konfiguračního souboru a společná pro všechny hry. Instance ostatních tříd v této složce jsou členské proměnné této třídy. Jsou zde třídy *Land*, *Sea* a *Mountains*, odvozené z třídy *Region* a uchovávající údaje o provinciích, jež se nemění v průběhu hry: výnosy, výchozí armáda, popisek, tvar, barva, pravděpodobnost odtržení a zvláštní jednotky, verbovatelné v provincii (například slon nebo fénická loď). Dále se ve složce nachází třídy *Pedestrian*, *Elephant*, *Ship* a *OutlandishShip*, které dědí ze třídy *SoldierType* a reprezentují typy vojáků.

Graf hierarchie tříd pro data provincií:



Graf hierarchie tříd pro typy vojáků:



4.1.3 Data

Ve složce *Data* se nacházejí třídy pro data, jež reprezentují stav hry, a proto se v průběhu stále mění. Přesto se však jedná o neměnné datové typy a změna se provádí nahrazením jednoho objektu za jiný. Dvě nejdůležitější třídy jsou *Player* a *Province*, které reprezentují o hráče a provincii. Kromě třídy *Player* je zde také třída *PlayerIdentity*, jež obsahuje hráčovo jméno, pořadí na tahu, hru a údaj, zda se jedná o lidského hráče či algoritmus, tedy informace, které hráče jednoznačně identifikují a nemění se během hry. Třída *Player* následně obsahuje instanci třídy *PlayerIdentity*, instanci třídy *Settings*, protože metody třídy potřebují znát nastavení, počet peněz, heslo, údaje, zda hráč žije a zda je na tahu, a neměnný seznam prvků *IAction*, jež obsahuje hráčem prováděné akce, jež se uskuteční při ukončení tahu. Kód rozhraní *IAction*:

```
public interface IAction
{
    (Player, Provinces, IAction?) Perform(Player active, Provinces provinces, Settings settings);
}
```

Akce tedy podobně jako příkaz přebírá informace o stavu a vrací změněný stav. Třetí vrácená hodnota je potom akce, která v příštím kole nahradí v poli akcí hráče tuto akci, nebo hodnota

null, pokud má být tato akce bez náhrady odstraněna. Rozhraní *IAction* implementují třídy *Loan* a *Manoeuvre*, které reprezentují splácení půjčky a přesun vojska. Třetí vrácená hodnota je u přesunu vojska vždy *null*, neboť vojsko se vždy na konci tahu přesune, zatímco u půjčky se vrací opět půjčka, jejíž dlužná částka je zvýšená o úrok a snížená o splacenou částku, nebo *null*, pokud již došlo ke splacení půjčky.

Třída *Province* obsahuje instanci třídy *Region*, jež uchovává základní a neměnné informace o provincii, a třídy *Settings*, která je využívána v metodách. Dále obsahuje *PlayerIdentity* hráče, jemuž provincie náleží, nebo hodnotu *null*, pokud provincie žádnému hráči nepatří. Důvod, proč třída *Province* obsahuje pouze identitu hráče a nikoliv přímo instanci třídy *Player*, je ten, že ostatní údaje o hráči se mohou měnit, což by při použití neměnných datových typů vyžadovalo pokaždé přímo měnit všechny provincie změněného hráče, což by do kódu vneslo zbytečnou složitost, vzhledem k tomu, že ve třídě *Province* je identita hráče použita pouze pro porovnání, zda hráč, který do provincie poslal vojenské jednotky je vládcem provincie, nebo se snaží provincii dobýt. Třída *Province* ještě obsahuje objekt třídy *Soldiers*, který reprezentuje současnou armádu v provincii a obsahuje neměnné pole typu *Regiment*, což je třída obsahující instanci *SoldierType* a počet vojenských jednotek onoho typu. Provincie jsou následně zapouzdřené ve třídě *Provinces*, která obsahuje pole provincií a objekt třídy *Graph*, v němž je uloženo, mezi kterými provinciemi vede cesta.

Mezi další třídy v této složce patří *Game*, obsahující informace o stavu samotné hry, což je údaj o tom, zda hra již začala nebo skončila (a kdy se tak stalo), zda je nově vytvořená, nebo zda brzy začne (a kdy se tak stane). Síla hráče v daném kole se ukládá do třídy *Power* a skládá se z počtu jeho čílových zemí, peněz, výše příjmů, síly vojáků a údaje, zda je hráč vůbec naživu. Hodnoty síly každého hráče pro dané kolo se ukládají do pole typu *Power* ve třídě *Powers*. Dále se ve složce nachází pomocné typy *Ratio*, jež reprezentuje zlomek od 0 do 1, a *Password*, jež reprezentuje hashované heslo. Třída *Brain* obsahuje algoritmus hráče robota.

4.2 Server

V projektu *Server* se nachází složky *Controllers*, *Files*, *Pages*, *Properties* a *Services*. Složky *Properties* a *Pages* slouží pro potřeby frameworku ASP.NET Core. Ve složce *Files* se nachází nastavení hry, načtená při spuštění do objektu *Settings*, a SQL kód pro vytvoření tabulek v databázi. Ve složce *Controllers* jsou třídy kontrolerů, jež jako parametry v konstruktoru dostanou objekty služeb, pomocí nichž získávají a zpracovávají data z databáze. Toho využívají k odpovídání na HTTP dotazy z klienta.

Složka *Services* obsahuje třídy služeb, které zajišťují práci s databází a se třídami z projektu *Shared*. Služby vždy implementují nějaké rozhraní a do metody ostatních tříd mají přístup jen k tomu rozhraní, aby se případná implementace mohla změnit. Připojení k databázi a provádění příkazů zaručuje rozhraní *IDatabase*, které je v současnosti implementováno třídou *SqliteDatabase* pro databáze Sqlite. Kdyby měla být v budoucnu nahrazena Sqlite databáze například databází MySQL, rozhraní by mohlo zůstat stejné a třídy, které databázi používají skrze něj, by zůstaly nedotčeny.

4.3 Client

V projektu *Client* se nachází složka *Pages*, v níž se nachází soubory Blazor komponentů. Každý takový soubor obsahuje kód v HTML i v C# a reprezentuje určitou část stránky i její chování.

Kód v C# může být spouštěn při načtení stránky nebo při interakci s uživatelem a provádí změny stránky tím, že může části HTML zobrazovat jen při splnění určité podmínky, vytvářet HTML kód v cyklu nebo vyjádřit část HTML kódu jako hodnotu výrazu v C# a při změně této hodnoty ji měnit. Navíc lze komponent použít jako HTML tag v jiném komponentu a tak rozdělit stránku do více souborů.

Ve složce *Services* se nachází služby, které může kód v C# využívat k posílání HTTP dotazů na server a přístupu k úložišti v prohlížeči a složka *Data* obsahuje datové typy, které definují strukturu formátu JSON používaného při komunikaci se serverem a zápisu do úložiště. Složka *Properties* obsahuje pouze konfigurační soubor *launchSettings.json* pro potřeby frameworku Blazor. Ve složce *wwwroot* se nachází soubory *favicon.ico* a *site.css*, které obsahují ikonu stránky a její CSS styly, a *index.html*, což je krátký HTML soubor, který se zobrazí při načítání stránky a pomocí JavaScriptu načte a spustí aplikaci.

Použitá literatura

[1] *Turn Based vs. Real Time Strategy*. URL: https://web.archive.org/web/20070226185919/http://www.strategyplanet.com/features/articles/pcp-turnvsreal/.

[2] *RTC design*. URL: https://web.archive.org/web/20070927142342/http://www.dunniwaydesign.com/rts_design.htm.

[3] A. Leff a J. T. Rayfield. "Web-application development using the Model/View/Controller design pattern". In: *Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference*. 2001, s. 118–127. DOI: 10.1109/EDOC.2001.950428.

[4] Mark Stefik a Daniel G. Bobrow. "Object-Oriented Programming: Themes and Variations". In: *AI Magazine* 6.4 (1985), s. 40. DOI: 10.1609/aimag.v6i4.508. URL: https://ojs.aaai.org/index.php/aimagazine/article/view/508.

[5] *Immutable objects in Java*. URL: https://repository.ubn.ru.nl/handle/2066/35735.

[6] Zhenjiang Hu, John Hughes a Meng Wang. "How functional programming mattered". In: *National Science Review* 2.3 (čvc. 2015), s. 349–370. ISSN: 2095-5138. DOI: 10.1093/nsr/nwv042. eprint: https://academic.oup.com/nsr/article-pdf/2/3/349/31566307/nwv042.pdf. URL: https://doi.org/10.1093/nsr/nwv042.