**STŘEDNÍ PRŮMYSLOVÁ ŠKOLA BRNO, PURKYŇOVA, PŘÍSPĚVKOVÁ ORGANIZACE**



**SPRÁVA ŠKOLNÍ KNIHOVNY**

**Petr Němec**

**V4C**

**Profilová část maturitní zkoušky**

**MATURITNÍ PRÁCE**

**BRNO 2020**

**Prohlášení**

Prohlašuji, že jsem maturitní práci Strategická hra pro OS Android vypracoval samostatně a použil jen zdroje uvedené v seznamu literatury.

Prohlašuji, že:

Beru na vědomí, že zpráva o řešení maturitní práce a základní dokumentace k aplikaci bude uložena v elektronické podobě na intranetu SPŠ Brno, Purkyňova, příspěvková organizace.

Beru na vědomí, že bude má maturitní práce včetně zdrojových kódů uložena v knihovně SPŠ Brno, Purkyňova, příspěvková organizace, dostupná k prezenčnímu nahlédnutí. Škola zajistí, že nebude pro nikoho možné pořizovat kopie jakékoliv části práce.

Beru na vědomí, že SPŠ Brno, Purkyňova, příspěvková organizace, má právo celou moji práci použít k výukovým účelům a po mém souhlasu nevýdělečně moji práci užít ke své vnitřní potřebě.

Beru na vědomí, že pokud je součástí mojí práce jakýkoliv softwarový produkt, považují se za součást práce i zdrojové kódy, které jsou předmětem maturitní práce, případně soubory, ze kterých se práce skládá. Součástí práce není cizí ani vlastní software, který je pouze využíván za přesně definovaných podmínek, a není podstatou maturitní práce.

Petr Němec

Troubsko-Veselka, Jihlavská 52

V Brně dne 17. 4. 2020 .........................................................

Vedoucí práce: RNDr. Lenka Hrušková

**Poděkování**

Děkuji vedoucímu maturitní práce RNDr. Lence Hruškové za metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé maturitní práce.

**Anotace**

Tato maturitní práce se zabývá vytvořením minimalistické strategické hry. Strategická hra je určena pro OS Android.

Cílem práce je popsat proces vytváření hry a případné nedostatky při jejím zhotovování a doručit funkční hru. Hra se hraje na trojúhelníkovém poli s figurami, které funkčně reprezentují kámen, nůžky a papír. Hráč bude nucen část figur rozdělit do skupin, přičemž platí, že různě velké skupiny jednotek mají různé vlastnosti. Část figur bude nucen dát samostatně nebo v rámci menších skupin, které budou vynikat větší pohyblivostí. Cílem hráče je vyvinout optimální strategii, díky které porazí veškeré jednotky protihráče nebo získá převahu. Převaha se vyhodnocuje při hře trvající více než dvacer kol.

# 

**Obsah**

[1 Teoretický úvod 8](#_Toc38458637)

[2 Seznam použitých zkratek 8](#_Toc38458638)

[3 Rozbor řešení 9](#_Toc38458639)

[3.1 Dominance jednotek 9](#_Toc38458640)

[3.2 Pravidla pro jednotky dle počtu jednotek ve skupině 10](#_Toc38458641)

[3.3 Přehled vlastností jednotek 10](#_Toc38458642)

[4 Vizualizace pohybu jednotek 10](#_Toc38458643)

[4.1.1 Jedna jednotka 11](#_Toc38458644)

[4.1.2 Dvě jednotky 12](#_Toc38458645)

[4.1.3 Tři jednotky 13](#_Toc38458646)

[5 Problémy spojené s vytvořením hry 14](#_Toc38458647)

[6 Problémy spojené s návrhem 14](#_Toc38458648)

[6.1 Původní objektový návrh aplikace 14](#_Toc38458649)

[6.2 Nelogické rozdělení vlastností do tříd 15](#_Toc38458650)

[6.3 Schéma tříd pro jednotky 16](#_Toc38458651)

[6.3.1 Chyba v návrhu Jednotek 16](#_Toc38458652)

[6.4 Problematické přistupování k vlastnostem objektu 17](#_Toc38458653)

[6.5 Řešení do nové verze projektu 17](#_Toc38458654)

[7 Návrh nové verze projektu 17](#_Toc38458655)

[8 UML diagram projektu 18](#_Toc38458656)

[8.1 Class diagram projektu 18](#_Toc38458657)

[8.2 Diagram spolupráce tříd při pohybu jednotek 20](#_Toc38458658)

[9 Adamova adresace 22](#_Toc38458659)

[9.1 Návrh souřadnicového systému 22](#_Toc38458660)

[9.2 Vzdálenost dvou bodů v trojúhelníkovém prostoru 23](#_Toc38458661)

[10 Co jsem ohlídal 24](#_Toc38458662)

[10.1 Null hodnoty 24](#_Toc38458663)

[Závěr 25](#_Toc38458664)

[Seznam ilustrací 26](#_Toc38458665)

[Seznam tabulek 27](#_Toc38458666)

[Zdroje 27](#_Toc38458667)

[Literatura 27](#_Toc38458668)

[Přílohy 28](#_Toc38458669)

# Teoretický úvod

Toto téma jsem si zvolil z toho důvodu, že mě vždy fascinovaly hry. Chtěl jsem také zjistit, jak těžké je nějakou hru navrhnout. Tato práce se snaží dokázat, že i jednoduchá hra s jednoduchými pravidly může být zábavná.

Při tvorbě zadání bylo třeba se oprostit od velkých her. Pokusil jsem se navrhnout hru tak, aby byla proveditelná a zároveň hratelná. Nakonec jsem se rozhodl pro strategickou hru. Základní prvky těchto her jsou velmi jednoduché a nevyžadují žádné velké grafické a sound designové nadání nebo zkušenosti, aby mohly fungovat.

Moje strategická hra původně obsahovala několik druhů jednotek a jako koncept se podobala zjednodušené verzi Age of empires. Tento nápad byl pochopitelně absolutně neproveditelný. Rozhodl jsem se problém zjednodušovat do doby, dokud jsem nedospěl k tomu, že již další zjednodušení není možné nebo by poškodilo hratelnost.

# Seznam použitých zkratek

C # - Programovací jazyk, ve kterém jsem vytvořil scripty komponent.

# Rozbor řešení

Hru jsem navrhl tak, že se bude skládat z pole trojúhelníků. Pole z trojúhelníku jsem zvolil, jelikož není tak obvyklé jako čtvercové nebo hexanové. Trojúhelník je také vhodný z toho důvodu, že ve hře jde o překrývání jednoteka v trojúhelníkovém poli se pohybuje pouze do tří směrů. Díky tomu je pravděpodobné, že hráči budou intuitivně vědět, jak táhnout, aby efektivně překryli protihráčovi jednotky.

Ve hře jsou tři druhy jednotek. Hráči na začátku hry rozmístí jednotky. Každá z jednotek zastupuje jednu z funkcí: kámen, nůžky nebo papír. Hráči vloží do herní plochy pět jednotek dle výběru a vytváří skupiny jednotek.

V průběhu hry mohou hráči přidat nové jednotky jen za předpokladu, že vytvoří skupinu dvou jednotek. Jednotky lze seskupit jen v případě, že jsou ve vzdálenosti maximálně dvou trojúhelníků (vzdálenost přes dvě hrany). Velikost hrací plochy je 26 x 26 trojúhelníků.

## Dominance jednotek

Přehled toho, která jednotka zvítězí nad kterou.

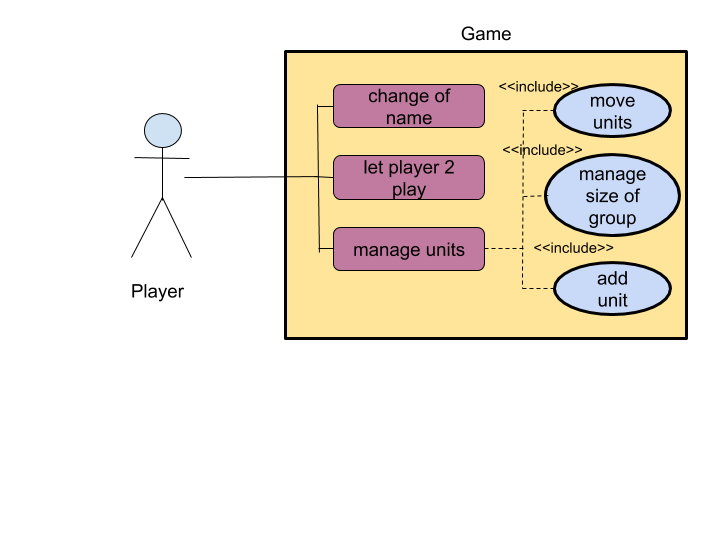
Věž zvítězí nad štítem, šíp zvítězí nad hradem a štít zvítězí nad šípem.

Obrázek - převaha jednotek

## Pravidla pro jednotky dle počtu jednotek ve skupině

Každý hráč má na začátku kola šest tahů. Různě velké skupiny jednotek mají různou spotřebu tahů a různé benefity. Tento model jsem vytvořil, aby byla velká variabilita hry. Vlastnosti jednotek lze opravit jen tím, že se hra bude hrát. Na základě zpětné vazby a zkušeností hráčů lze upravit vlastnosti jednotek nebo velikost hrací plochy.

## UML diagram projektu

Hráč může měnit vlastnosti jednotek (velikost, polohu, počet) nebo nechá hrát druhého hráče. Na začátku hry si může změnit jméno, aby bylo jasné, kdo je na tahu.

Obrázek 11 - UML diagram hry

## Přehled vlastností jednotek

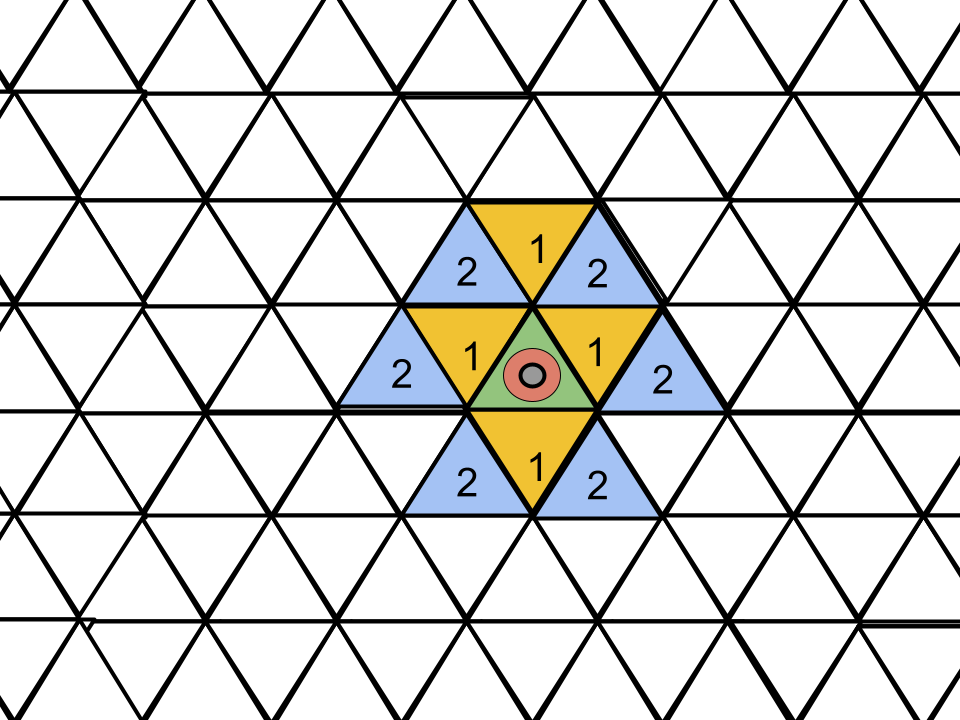
Sloupec „Počet jednotek“ značí počet jednotek ve skupině. „Počet tahů“ značí počet tahů, který může jednotka vykonat v daném kole. „Speciální benefity“ jsou speciální vlastnosti dané skupiny jednotek.

Tabulka - vlastnosti různě velkých skupin jednotek

|  |  |  |
| --- | --- | --- |
| **Počet jednotek** | **Počet tahů** | **Speciální benefity** |
| jedna | 6/poč. jednotek |  |
| dvě | 6/poč. jednotek | Spawn jednotek  - 2 tahy |
| tři | (6/poč. jednotek) + 2 | + 2 tahy |
| tři a více | (6/poč. jednotek) + 2 | + 2 tahy |

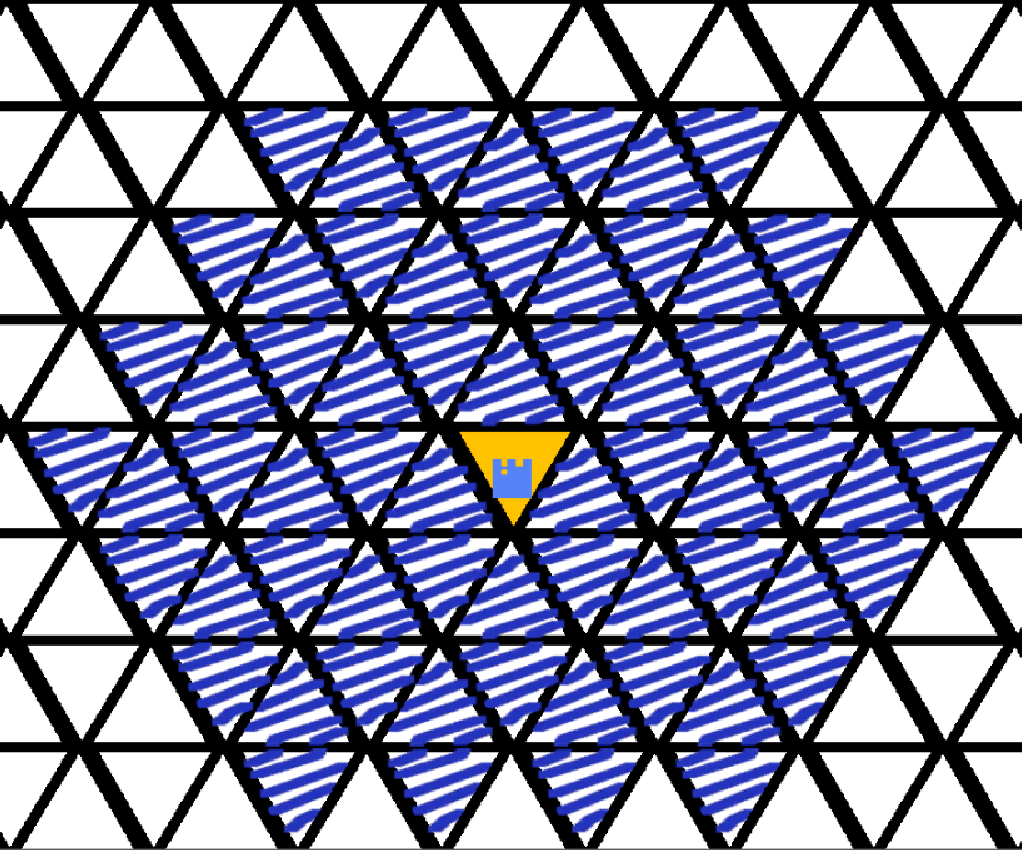
# Vizualizace pohybu jednotek

Jednotka se může pohybovat pouze přes hrany trojúhelníku. Jeden přechod hrany se rovná jednomu kroku jednotky.

Čísla značí vzdálenost, která se odečte od celkového počtu tahů. Na zeleném políčku v centru útvaru se nachází naše jednotka.

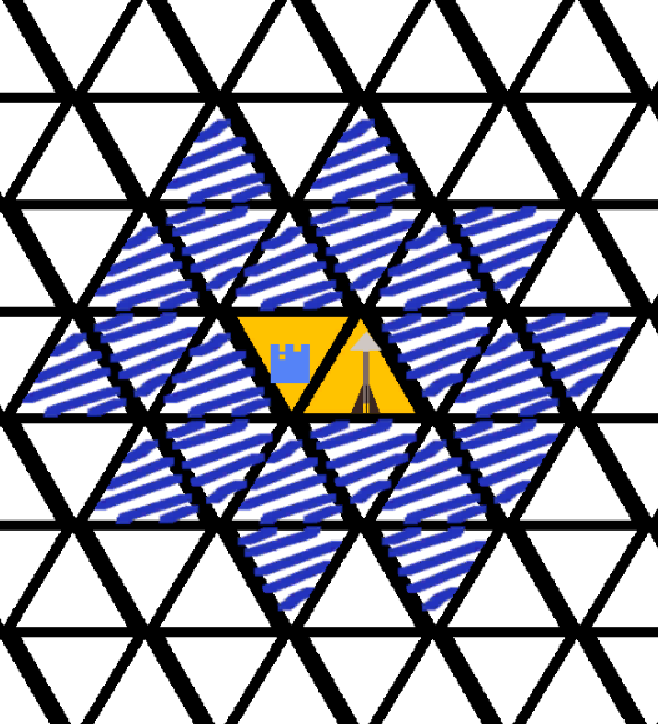
Obrázek - popis odečítání vzdálenosti

### Jedna jednotka

Jedna jednotka má největší pohyblivost šest políček. Její nevýhoda je v tom, že existuje šance šedesát šest procent, že potká jednotku, která ji porazí.

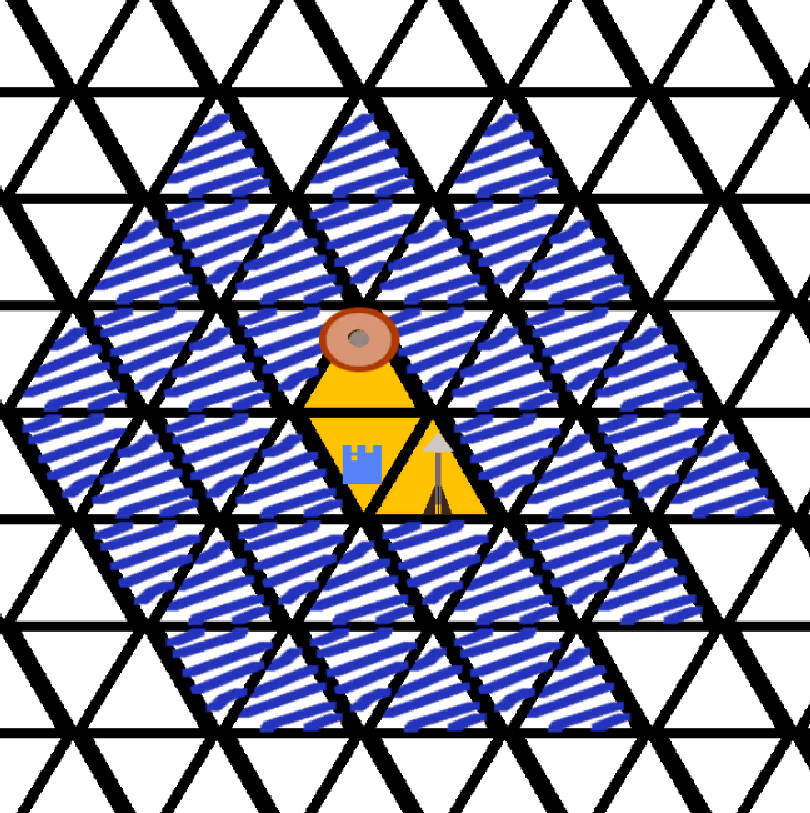
Obrázek - vzdálenost, kterou ujde jedna jednotka

### Dvě jednotky

Dvě jednotky se pohybují o tři trojúhelníky. Jejich speciální schopností je, že během hry je možné k nim přidat jednu jednotku za cenu tří tahů.

Obrázek - vzdálenost, kterou ujdou dvě jednotky

### Tři jednotky

Tři jednotky se pohybují o čtyři trojúhelníky. Jejich výhodou je to, že se dají nakombinovat tak, že skupina má všechny druhy jednotek.

Obrázek - vzdálenost, kterou ujdou tři jdednotky

# Problémy spojené s vytvořením hry

Problémy jsem měl především s návrhem hry jako takové. Poté se vyskytly problémy s referencemi. Očekával jsem, že se GameObjecty uložené ve hráči budou chovat jako reference na objekty uložené v MapGeneration script. Také jsem čelil tomu že jsem mnohé funkce a třídy navrhl zbytečně složitě, je to dáno především nulovými předchozími zkušenostmi.

# Problémy spojené s návrhem

S návrhem a realizací jsem začal již o prázdninách. Největší můj problém byla nejistota, zda se jedná o dobrou hru a zda jsou pravidla odpovídající. S tímto pocitem se opravdu špatně pracovalo na základních funkcích hry. Proto jsem se rozhodl, že pravidla sepíši a budu přemýšlet o jednotlivých aspektech. Nejdůležitějším bodem úvah pro mě bylo, zda je pravidlo funkčně nutné a zda se nejedná o podvědomou inspiraci od nějakého velkého titulu. Snažil jsem se držet zásady, že hra by měla být originální, neměla by být vystavena na kostře hry jiné a měla by obsahovat jen ty prvky, které pro ni mají smysl.

Nakonec jsem pravidla sepsal do té formy, jak jsou uvedena v úvodu této práce. Netvrdím, že jsou perfektní, ale po takovém čase stráveném nad jejich minimalizováním jsem si řekl, že tato pravidla jsou zadáním a nesmím se nad nimi již hlouběji zamýšlet. Tento krok jsem zvolil z toho důvodu, abych se mohl věnovat programování hry jako takové.

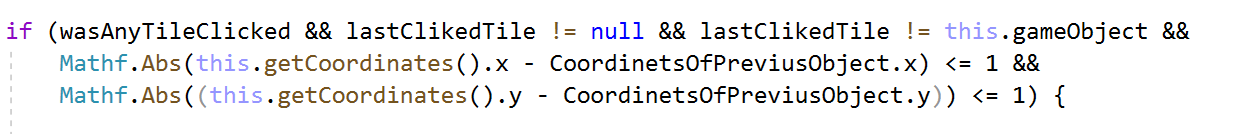
## Původní objektový návrh aplikace

Základní chybou, kterou se mi podařilo udělat, bylo to, že jsem se nezamyslel dostatečně nad třídami, které by měla aplikace obsahovat. Příkladem je chybějící třída na zpracování herní logiky (GameLogic) tato třída by obsahovala funkce, které prosazují herní pravidla jako například: maximální vzdálenost, na kterou se lze pohnout, konstanty typu maximální počet tahů hráče atp.

## Rozdělení vlastností do tříd

*MaximalNumberOfMoves* by měl patřit právě do třídy GameLogic. 

Obrázek - přehled zákládních třid

Jako správně navrženou třídu považuji TileClick. Ten obsahuje odkaz na jednotku a ta odkaz na hráče, který ji položil. Její hlavní vadou je vyhodnocování herní logiky, jejíž funkce měli být definovány ve třídě GameLogic. To, že jsem to tímto způsobem neudělal, mělo za následek nabobtnalou třídu a nepřehledné oblasti kódu jako je tento.

Obrázek - nepřehledný kód

Tato část kódu slouží k pohybu do vzdálenosti 1 herního políčka.

## Schéma tříd pro jednotky

Obrázek - schema třid jednotek

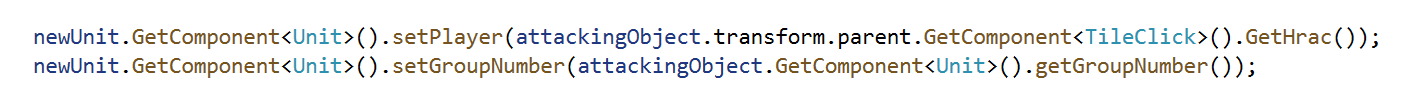
Základní třídou je třída Unit. Třídy „Stit“, „Sip“, „Vez“ (která na schématu chybí), pouze rozšiřují třídu Unit o svůj datový typ, který se používá ve funkci *unitFight()* na vyhodnocení vítězství souboje jednotek. Konkrétní jednotky „Stit“, „Sip“, „Vez“se liší ještě texturou. Textura se nastavuje přímo v unity editoru. Není tedy přítomna v přehledu tříd.

### Chyba v návrhu Jednotek

Proměnná *coordinatesOfunit* jezbytečná. V Unity jsou souřadnice dostupné z *this.transform.position*, takže je zbytečné mít extra proměnou pro přístup k souřadnicím. Tyto souřadnice mají tu výhodu, že se počítají vzhledem k pivotu. Díky nim je možné jednoduše přemisťovat jednu jednotku na jinou. [27]

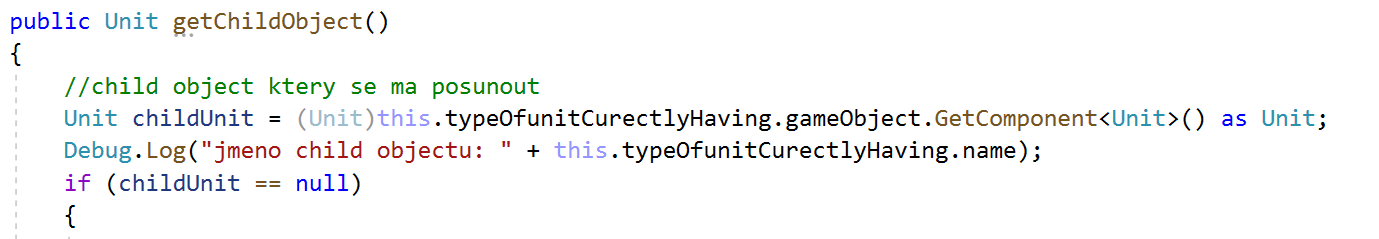
*UnitFight()* by se měl logicky nacházet spíše ve třídě GameLogic než ve třídě Unit.

## Problematické přistupování k vlastnostem objektu

Unity je specifické tím, že každý GameObject má velké množství atributů, mezi kterými lze i cyklicky přepínat. Hlavní problém, který tím vzniká, jsou dlouhé řádky přistupující ke konkrétnímu atributu. To vedlo k poměrně nepřehledné situaci a dlouhým minutám studia.

Obrázek - nepřehledný přístup k vlastnostem objektů

## Řešení do nové verze projektu

Řešení, které mě bohužel napadlo až nyní, jsou jednoduché funkce, které budou mít i nadále jako vstup GameObject. Vždy v začátku funkce vytvořím zástupnou proměnou, do které uložím to, s čím chci pracovat.

Obrázek - řešení nepřehledného přístupu k vlastnostem objektů

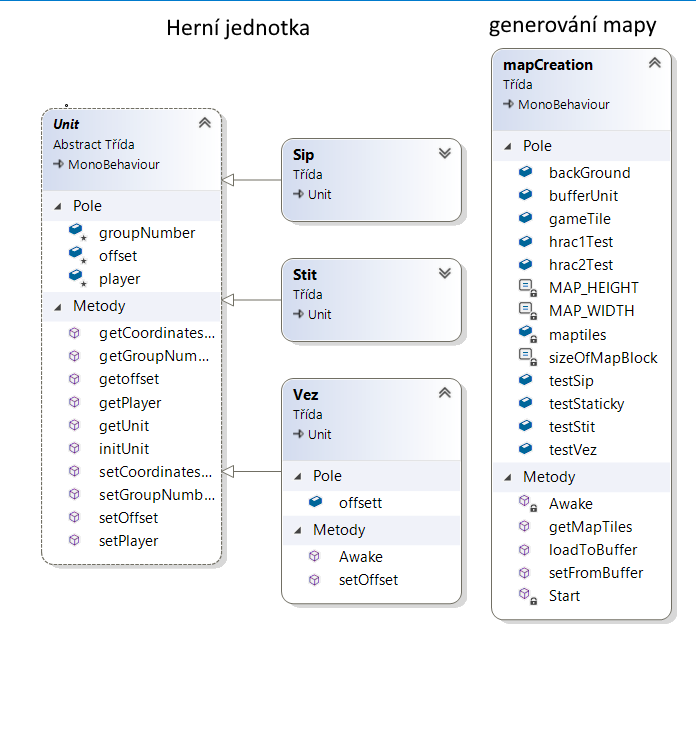
# Návrh nové verze projektu

Nový návrh zčásti řeší všechny problémy spojené s předchozí verzí aplikace. Problémy jsou nepřehlednost tříd a nepřehlednost kódu. V novém návrhu se také budu snažit zredukovat počet řádků na třídu.

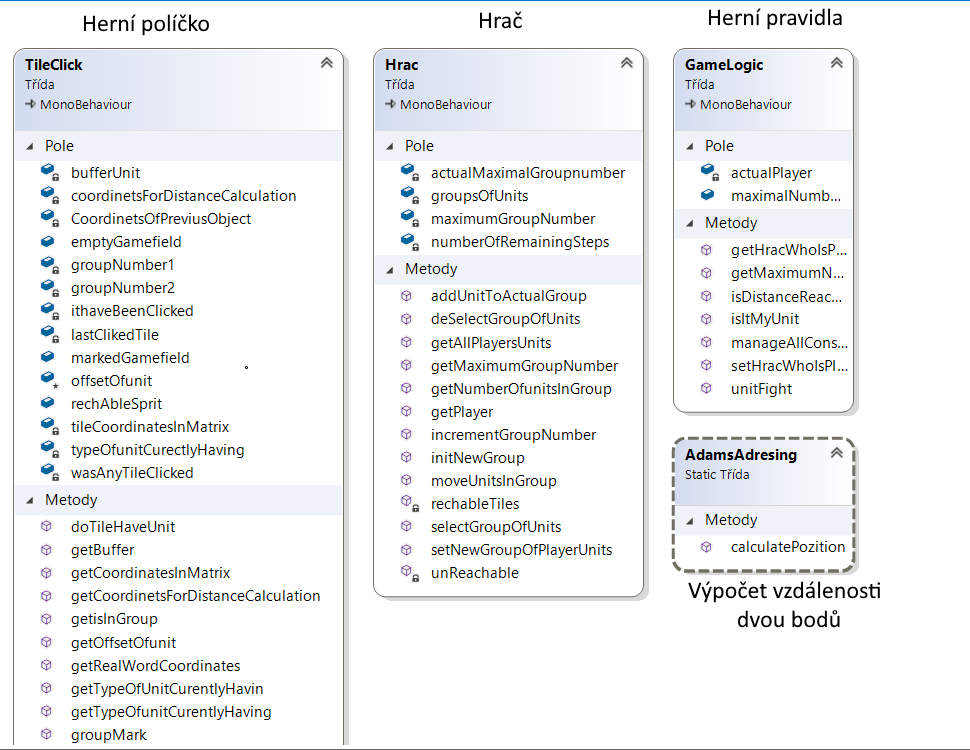
//zmenim az to bude hotové

Poslední komit ze staré verze je v branch *refactoring* bd2140788c191ed5. V nové verzi projektu pokračuji z branch *refacotirng.* Tu se mi snad povede rebasenout na master větev.

# Class diagram projektu

Herní jednotky dědí ze třídy Unit. Speciální typy jednotek využívám pouze k vyhodnocení vítěze souboje. Abych poznal, která jednotka, proti které bojuje (např: „Stit“ proti „Vezi“). Herní jednotky žádné speciální vlastnosti nemají kromě offsetu, který by měl sloužit k vycentrování spritu.

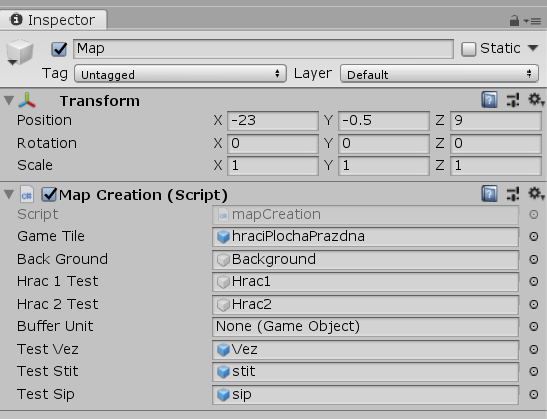
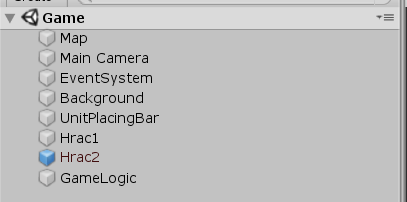
Obrázek - class diagram nového projektu



Obrázek - class diagram ostatních komponent

MapCreation generuje mapu, tu vyplňuje herními poličky (Tileclick). Herní políčko reaguje na onClickEvent() a tím dává do „pohybu” herní jednotky. Při testování se ze třídy mapCreation vytváří i bojové jednotky Unit. Tyto jednotky se uloží spolu s tilem (herním políčkem), na kterém leží, do třídy „Hrac“. Tím pádem je jasné, který hráč má které jednotky. Tyto třídy jsou inicializovány přes třídu generování mapy a jsou vloženy do proměnných přes inspektor.

Prefab Map v sobě obsahuje script MapCreation. Do něj jsem přes public proměnné vložil manuálně vytvořené prefaby. Modrá barva prefabu značí, že se jedná o kopii, která je linkem na originál. Změním-li originální prefab, změní se i kopie.

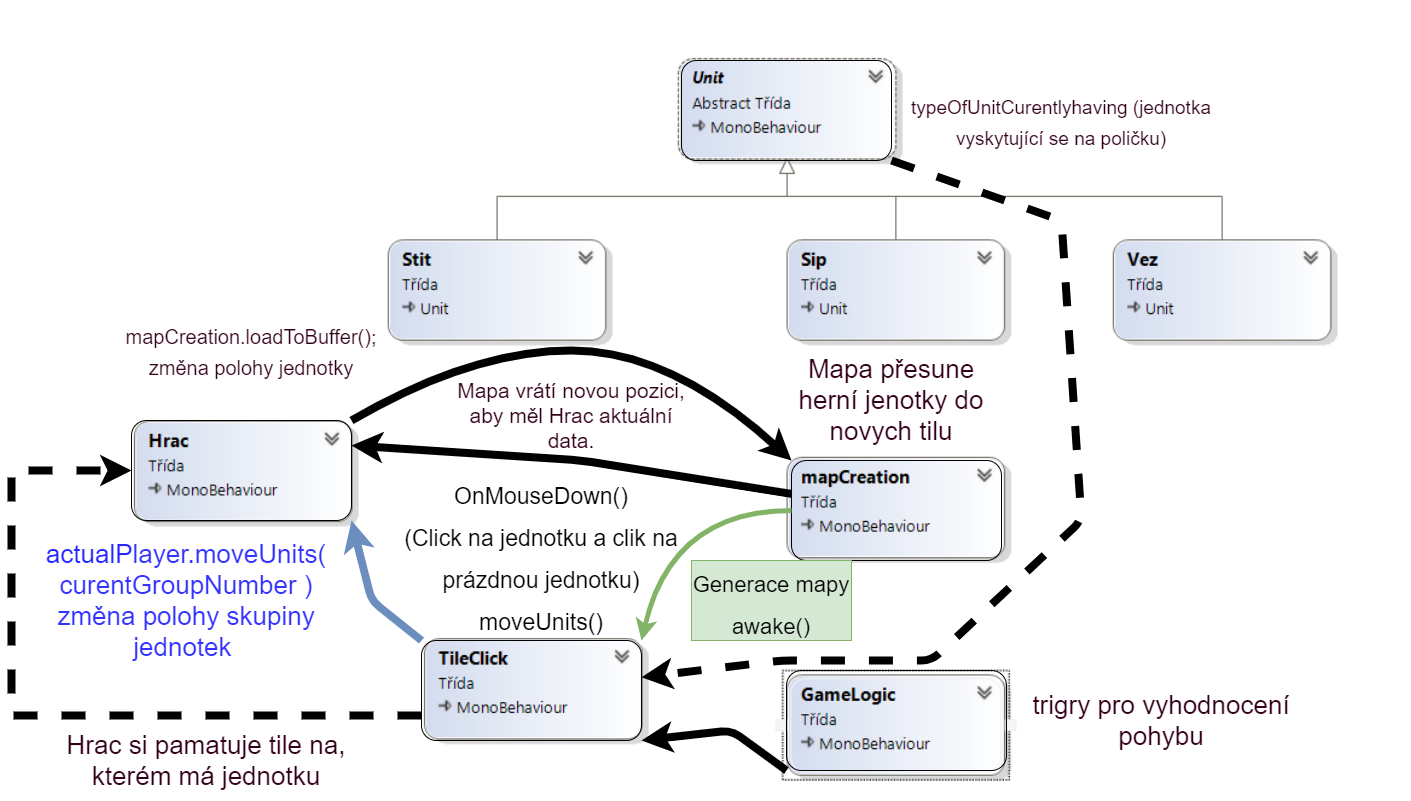
Hiearchie projektu Inspector komponenty (Map)

Obrázek - hiearchie projektu

Obrázek - UNITY inspektor

## Diagram spolupráce tříd při pohybu jednotek

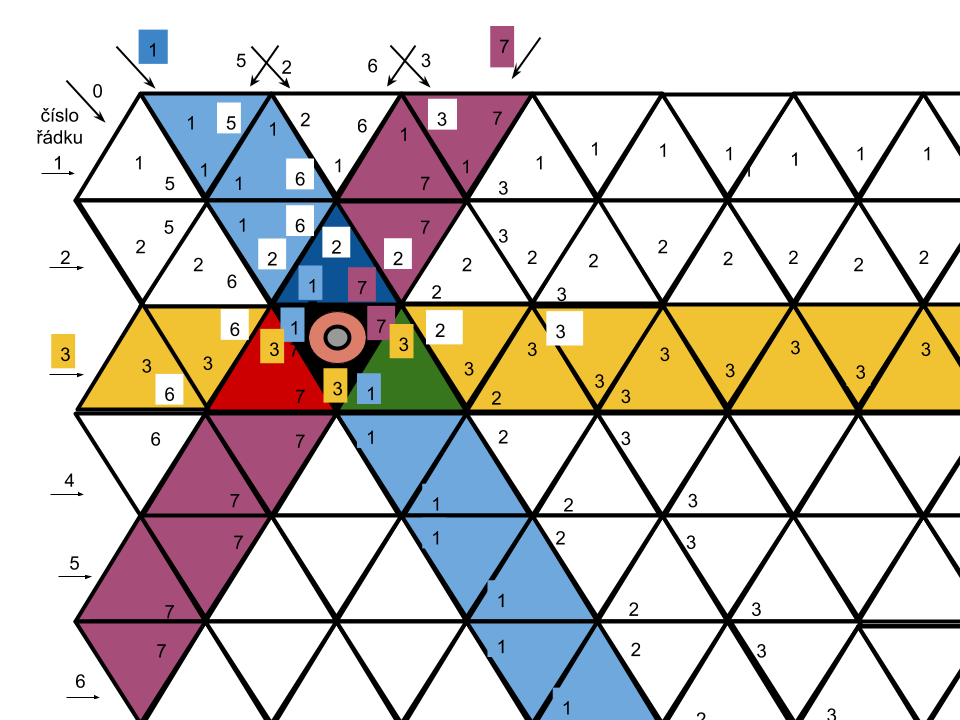
Po vytvoření mapy se na herní plochu přidají herní polička (TileClick). Poté se na konkrétní políčka přidají jednotky patřící konkrétním hráčům. Při kliknutí na tile (Tile Click) se aktivuje OnMouseDown() event. Za předpokladu, že klineme na tile (TileClick) s jednotkou aktuálního hráče a poté na prázdnou jednotku v dosahu, uskuteční se pohyb. Třída „GameLogic“ si pamatuje aktuálně hrajícího hráče a pravidla ohledně velikosti skupiny a maximální vzdálenosti, kterou je schopna jednotka urazit.

Při pohybu se vezme naposledy kliklá jednotka a získá se číslo její skupiny. Číslo skupiny je Y souřadnice v poli groupsOfUnits ve třídě „Hrac“. Tímto způsobem se zaktivuje celá skupina. Tato skupina se poté „teleportuje” na novou souřadnici. „Hrac“ si pamatuje „List“ <List< TileClick>> třídu a ne list jednotek (Unit) jako takový. Proto je po pohybu třeba vrátit z mapCratin cílovou jednotku pro aktualizaci skupiny jednotek v „Hraci“. Tím se pochopitelně návrh dost zkomplikoval. Příště se pokusím problém vyřešit tak, že budu ukládat jen základní třídy. Nezkoušel jsem to, ale poté by se mělo jednat o referenci, a ne o nový objekt.

Obrázek - diagram spolupráce tříd

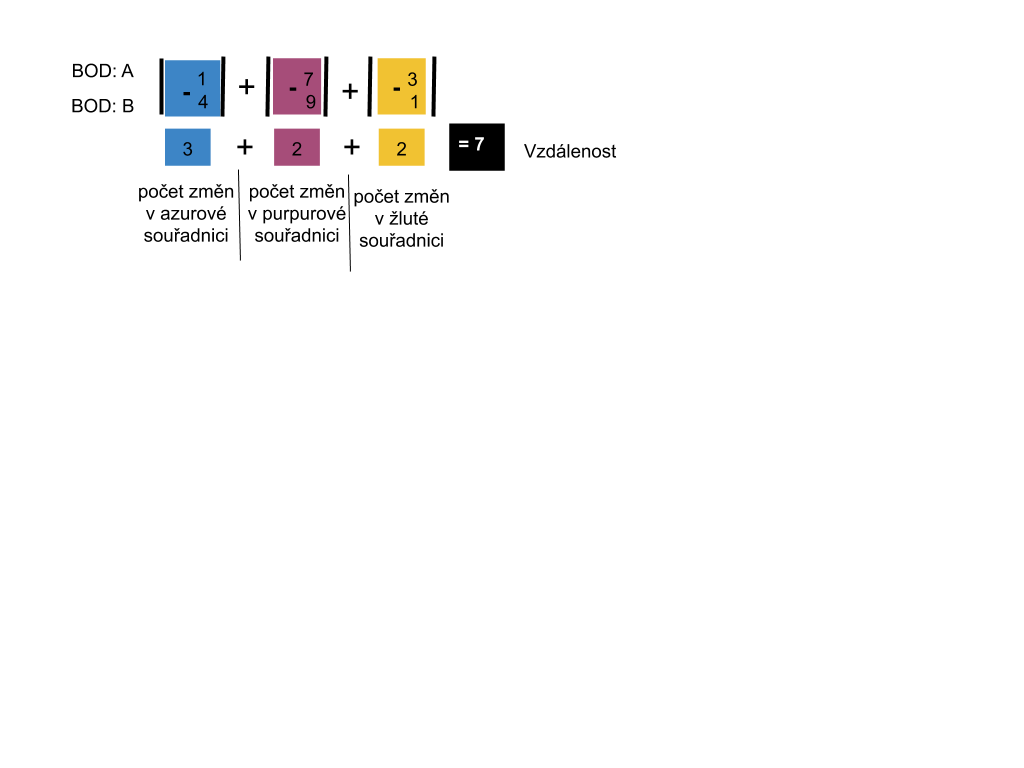
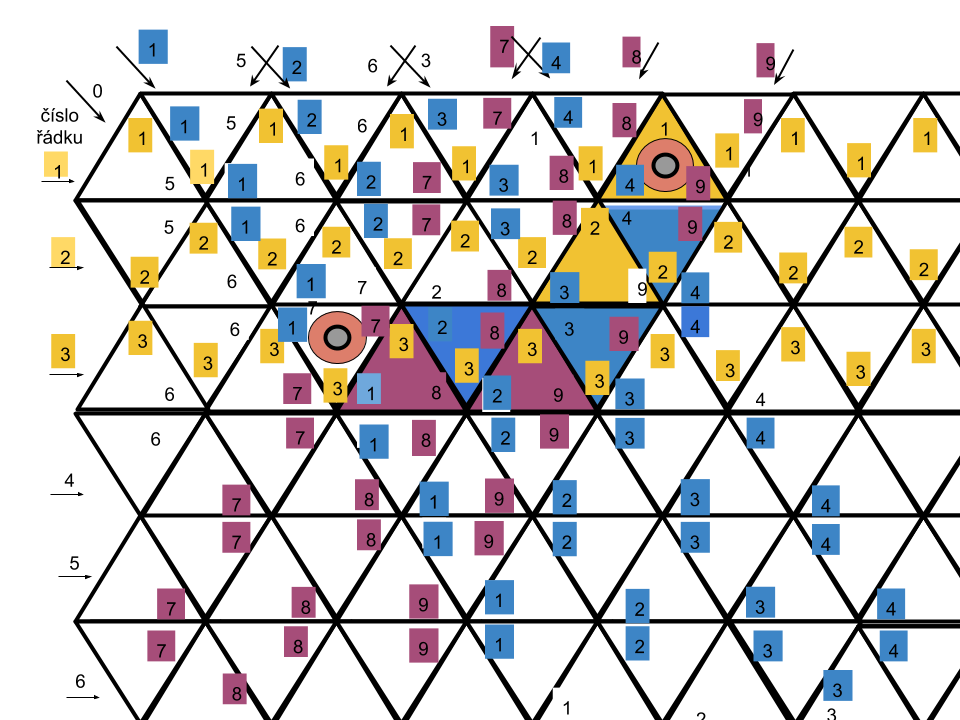
# Adamova adresace

## Návrh souřadnicového systému

Adam spolu se mnou navrhnul nejdříve souřadnicový systém popisující osy, ve kterých se může jednotka pohybovat. Tento souřadnicový systém se skládá ze tří souřadnic. Diagonála z levého horního do pravého dolního rohu ( **\** ) je pro náš bod označena azurově a má číslo jedna. Veškeré tyto diagonály mají vždy přiřazeno číslo o |1| odlišné od předchozí hodnoty. Každý bod je také označen purpurovou souřadnicí z pravého horního rohu do levého dolního rohu ( / ). Kombinací purpurové diagonály s azurovou diagonálou vznikne dvojice tmavě modrých bodů na řádku 2 a 3. Abychom věděli, na kterém z těchto řádku se bod nachází, tak je každému bodu přiděleno i číslo řádku označené žlutě. Tyto tři souřadnice purpurová, azurová a žlutá tvoří jeden černý bod který představuje souřadnice naší jednotky (jednoho konkrétního trojúhelníku).

Obrázek - trojúhelníkový souřadnicový systém

## Vzdálenost dvou bodů v trojúhelníkovém prostoru

Bod A (1; 7; 3), bod B (4; 9; 1). Souřadnice jsou na tomto příkladu ve formátu (azurová, purpurová, žlutá). Vzorec sčítá počet přechodu mezi jednotlivými souřadnicemi (barvami). Znění vzorce je:|A azurová – B azurová| + | A purpurová – B purpurová | + | A žlutá – B žlutá | = vzdálenost.

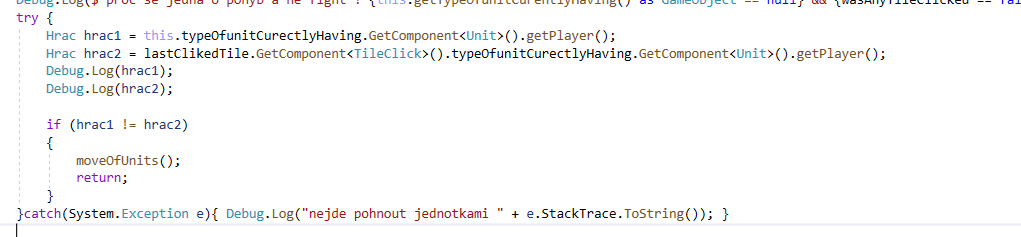
Obrázek - vysvětlení vzorce na vzdálenost dvou bodů

Obrázek - rovnice na výpočet vzdálenosti dvou bodů

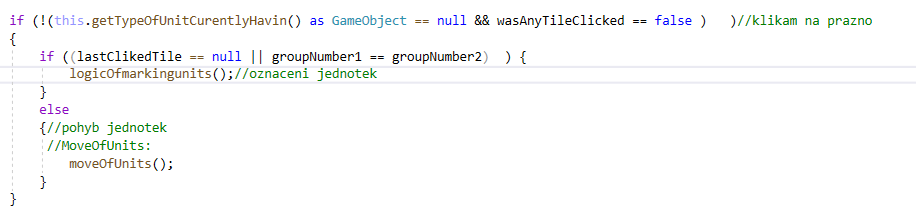
# Co jsem ohlídal

Klasickou věcí, kterou by každý čekal, že byla třeba hlídat, byl pohyb jednotek mimo rozsah pole. To ale nebylo třeba, jelikož mimo rozsah pole nejsou žádné jednotky, na které by se dalo kliknout a nedá se tam tím pádem ani pohnout. Ale i tak zbyly některé věci, na které si bylo třeba dát pozor.

## Null hodnoty

Obrázek níže zobrazuje výjimku. Tato výjimka konkrétně je nežádoucí, jelikož se nachází v částí kódu, ve které by typeOfunitCurentlyHaving ani lastClicked tile neměl mít žádnou z vlastností null. K této výjimce pochopitelně nemá nikdy dojít, ale proč může.

Obrázek - ohlídání null, hodnot

K výjimce může dojít, protože většina políček (TileClick) nemá přiřazenou jednotku. V případě, že nemá jednotku přiřazenou, jedná se o prázdné políčko a dá se na něj pohnout. Proto jsem této vlastnosti využil k tomu, abych zjistil, zda se dá na tileclik dostat. Pádu aplikace jsem zamezil použitím klíčového slova „AS“, které umožňuje pracovat s nulovou hodnotou.

Obrázek - práce s null hodnotou

# Závěr

* shrnutí celé práce
* k jakým výsledkům a závěrům autor došel
* jak moc se podařilo splnit dané zadání
* vlastní zajímavé postřehy z celkového řešení problému

# Seznam ilustrací

[Obrázek 1 - převaha jednotek 4](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713234)

[Obrázek 11 - UML diagram hry 5](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713235)

[Obrázek 2 - popis odečítání vzdálenosti 6](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713236)

[Obrázek 3 - vzdálenost, kterou ujde jedna jednotka 7](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713237)

[Obrázek 4 - vzdálenost, kterou ujdou dvě jednotky 8](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713238)

[Obrázek 5 - vzdálenost, kterou ujdou tři jdednotky 9](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713239)

[Obrázek 6 - přehled zákládních třid 11](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713240)

[Obrázek 7 - nepřehledný kód 11](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713241)

[Obrázek 8 - schema třid jednotek 12](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713242)

[Obrázek 9 - nepřehledný přístup k vlastnostem objektů 13](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713243)

[Obrázek 10 - řešení nepřehledného přístupu k vlastnostem objektů 13](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713244)

[Obrázek 12 - class diagram nového projektu 14](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713245)

[Obrázek 13 - class diagram ostatních komponent 15](#_Toc38713246)

[Obrázek 15 - hiearchie projektu 16](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713247)

[Obrázek 14 - UNITY inspektor 16](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713248)

[Obrázek 16 - diagram spolupráce tříd 17](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713249)

[Obrázek 17 - trojúhelníkový souřadnicový systém 18](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713250)

[Obrázek 18 - vysvětlení vzorce na vzdálenost dvou bodů 19](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713251)

[Obrázek 19 - rovnice na výpočet vzdálenosti dvou bodů 19](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713252)

[Obrázek 20 - ohlídání null, hodnot 20](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713253)

[Obrázek 21 - práce s null hodnotou 20](file:///C:\Users\petrn\OneDrive\Documents\SimpleStrategicGameTryangles\mp\dokumentace-Try(different)angle-Petr-Němec.docx#_Toc38713254)

# 

# Seznam tabulek

[Tabulka 1 - vlastnosti různě velkých skupin jednotek 5](#_Toc38460552)

## Zdroje

## Literatura

**Změna polohy objektu**

Civilization/Dungeon Tile Movement & Pathfinding #3 - YouTube. YouTube [online]. Dostupné z: <https://www.youtube.com/watch?v=0Q2z5g28ows>

<https://docs.unity3d.com/ScriptReference/Transform-position.html>

**Podrobnosti k ovládání a vlastnostem unity**

Unity - Manual: Transform. [online]. Copyright © 2020 Unity Technologies. Publication Date [cit. 19.04.2020]. Dostupné z: https://docs.unity3d.com/Manual/class-Transform.html



# Přílohy

1. Odkaz na github maturitní práce: <https://github.com/PetrNemecBellator/maturitni-prace/network>