|  |
| --- |
|  |
| Spojená škola, Komárňanská 28, Nové Zámky  s o. z. Stredná priemyselná škola elektrotechnická S. A. Jedlika - Jedlik Ányos Elektrotechnikai  Szakközépiskola a Obchodná akadémia -   * Kereskedelmi Akadémia |
|  |
| 3D PC HRA |
|  |
| MICHAL ONDREJÓ |
|  |
| 2014 |
|  |

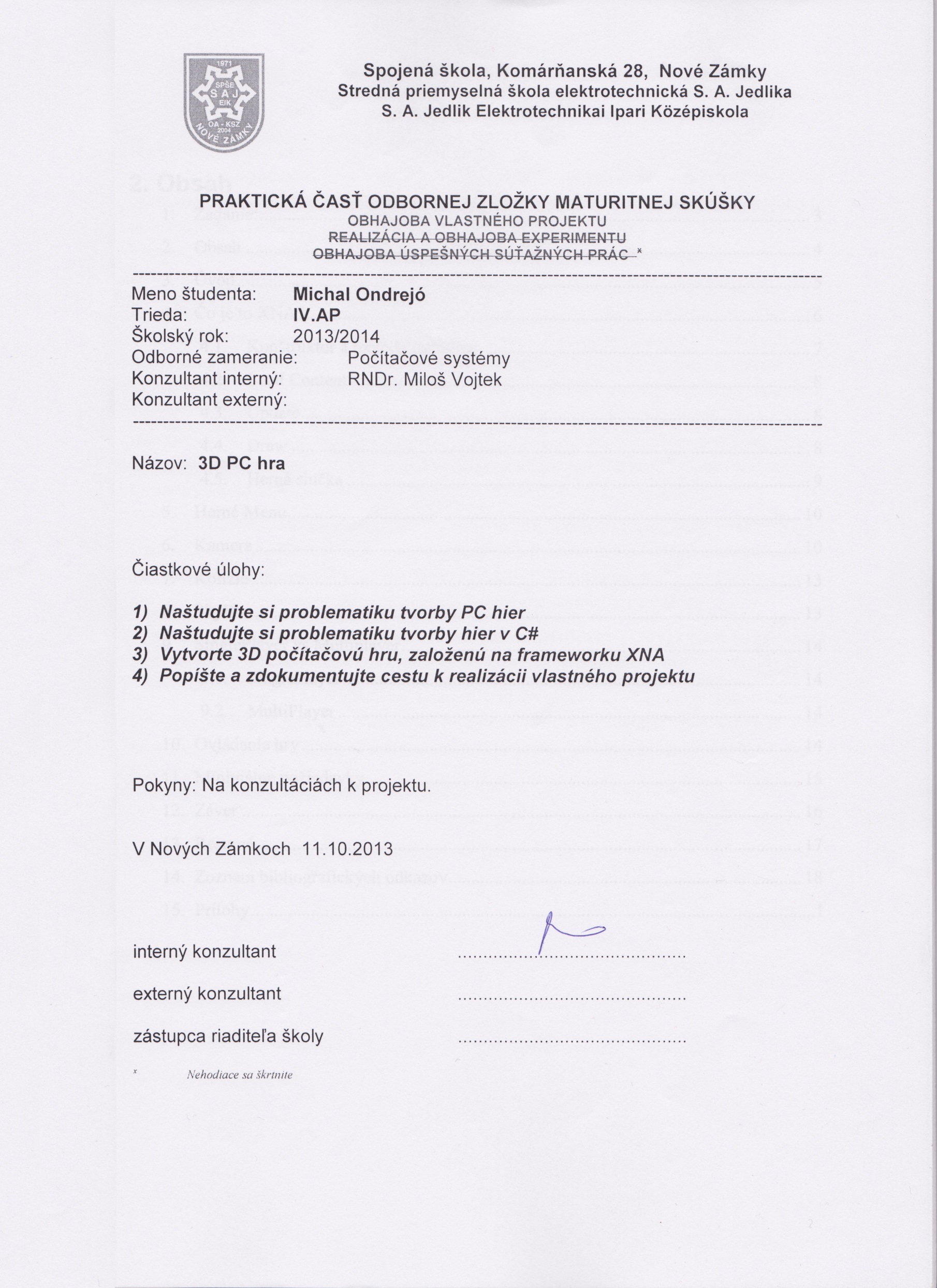
|  |
| --- |
|  |
| 3D PC HRA |
| VLASTNÝ PROJEKT |
| MICHAL ONDREJÓ |
| Študijný odbor:Elektrotechnika |
| Konzultant: RNDr.Miloš Vojtek |
|  |
| Nové Zámky 2014 |

**Čestné vyhlásenie**

Ja, dolupodpísaný Michal Ondrejó, študent 4. AP triedy Strednej priemyselnej školy S. A.Jedlika v Nových Zámkoch, týmto vyhlasujem, že som túto prácu vyhotovil sám, s použitím uvedenej literatúry a podľa rád môjho konzultanta.

..............................................

Podpis



# Obsah

[1. Zadanie: **Chyba! Záložka není definována.**](#_Toc383202269)

[2. Obsah 5](#_Toc383202270)

[3. Úvod 6](#_Toc383202271)

[4. Čo je to XNA? 7](#_Toc383202272)

[4.1. Konštruktor a metóda Initialize 8](#_Toc383202273)

[4.2. Load Content 9](#_Toc383202274)

[4.3. Update 9](#_Toc383202275)

[4.4. Draw 9](#_Toc383202276)

[4.5. Herná slučka 10](#_Toc383202277)

[5. Herné Menu 11](#_Toc383202278)

[6. Kamera 12](#_Toc383202279)

[7. Kolízie 15](#_Toc383202280)

[8. SkyBox 15](#_Toc383202281)

[9. SinglePlayer vs Multiplayer 16](#_Toc383202282)

[9.1. SinglePlayer 16](#_Toc383202283)

[9.2. MultiPlayer 16](#_Toc383202284)

[10. Ovládanie hry 17](#_Toc383202285)

[11. Minimálne požiadavky 17](#_Toc383202286)

[12. Záver 18](#_Toc383202287)

[13. Resumé 19](#_Toc383202288)

[14. Zoznam bibliografických odkazov 20](#_Toc383202289)

[15. Prílohy I](#_Toc383202290)

# Úvod

Niečo o autorovi: Prvý počítač som dostal už na základnej škole ako 8 ročný. V tej dobe som sa začal hrať aj prvé počítačové hry. Ako asi každý hráč, vždy ma zaujímalo, ako sa počítačové hry tvoria, keďže som programovať nevedel. Skúšal som rôzne herné enginy od rôznych firiem, ale keďže som nevedel presne čo a ako, tak som s tým prestal.

Programovať som sa naučil až keď som nastúpil na strednú školu a to v jazyku C++. Programovanie ma hneď začalo baviť. Prvé programy boli síce len konzolové, ale aj tak som sa snažil vytvoriť, sen asi každého počítačového hráča, vlastnú počítačovú hru. Prvá hra bola jednoduchá konzolová hra, loto, v ktorej si hráč zvolil číslo od 0 do 10 a ak sa zadané číslo rovnalo číslu, ktoré sa náhodne vygenerovalo, hráč dostal body, a keď sa nerovnali, body boli hráčovi odobrané. V 3. Ročníku som sa zoznámil s programovacím jazykom C#, v ktorom sme sa naučili, ako vytvárať prvé grafické aplikácie. Keď som sa v škole dozvedel, že ako maturitnú prácu môžem vytvoriť počítačovú hru, nadchlo ma to a zisťoval som si ďalšie informácie, ako sa dajú vytvárať hry.

Zistili sme, že na tvorbu 3D PC hier sa dajú použiť rôzne frameworky. Napr.: OpenGL, DirectX, XNA. Vtedy sme sa zoznámili s frameworkom XNA, pomocou ktorého sme sa najprv zoznamovali s 2D grafikou, a neskôr s 3D grafikou, pomocou rôznych návodov, nájdených na internete až to vyvrcholilo k vytvoreniu mojej prvej 3D hry.

Cieľom tejto práce, je teda vytvorenie funkčnej počítačovej hry, s možnosťou hrania proti počítaču (singleplayer) a tiež s možnosťou hrania proti reálnemu človeku (multiplayer).

V tejto práci sa najprv budeme zaoberať tým, čo vlastne XNA je a ako vlastne funguje. Potom si priblížime niekoľko základných prvkov hry, kde si spomenieme pár vecí čo je možné vidieť a potom tie, ktoré sú nášmu zraku skryté, ako kolízie a základné možnosti hry - singleplayer a multiplayer.

# Čo je to XNA?

Možno ste sa stretli s pojmom DirectX. Je to sada knižníc, ktoré umožňujú priame ovládanie grafickej karty. Dosiahne sa tak maximálny výkon a túto technológiu využívajú všetky moderné počítačové hry. Ako to už býva, vysoký výkon je bohužiaľ často spojený s nízkou úrovňou programovania. Priama práca s DirectX je problémová a je väčšinou realizovaná cez C++, teda nízky jazyk s manuálnou správou pamäte. Vďaka tomu je vývoj zložitý a ťažkopádny. S príchodom technológie .NET a moderného jazyka C# prišiel Microsoft s novou verziou DirectX, tzv. Managed DirectX (managed ako riadený, tým je na mysli najmä automatická správa pamäte). Managed DirectX dokázalo využiť až 98% výkonu DirectX a zároveň umožňovalo vysokoúrovňovú prácu. Nakoniec z neho vzišlo XNA, ako vysokoúrovňová nadstavba nad DirectX pre C# .NET, určená pre pohodlnú tvorbu profesionálnych hier.

XNA je framework, to znamená, že to nie je nejaký program alebo jazyk. Je to sada knižníc do jazyka C#. Vývoj hier v XNA prebieha vo Visual Studio. Vďaka tejto skutočnosti sa pri vývoji hier učíte C#, v ktorom je po programátoroch veľký dopyt. Na rozdiel od klikania v gamemakery, teda niečo naozaj viete a nemusíte sa báť o zamestnanie.

XNA má svoju špecifickú štruktúru:

namespace Projekt

{

**public** **class** Game1 : Microsoft.Xna.Framework.Game

{

GraphicsDeviceManager graphics;

SpriteBatch spriteBatch;

**public** Game1()

{

graphics = **new** GraphicsDeviceManager(**this**);

Content.RootDirectory = "Content";

}

**protected** override **void** Initialize()

{

base.Initialize();

}

**protected** override **void** LoadContent()

{

spriteBatch = **new** SpriteBatch(GraphicsDevice);

}

**protected** override **void** UnloadContent()

{

}

**protected** override **void** Update(GameTime gameTime)

{

**if** (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed)

**this**.Exit();

base.Update(gameTime);

}

**protected** override **void** Draw(GameTime gameTime)

{

GraphicsDevice.Clear(Color.CornflowerBlue);

base.Draw(gameTime);

}

}

}

## Konštruktor a metóda Initialize

Konštruktor - tu sa inicializuje premenná graphics a tiež sa nastaví koreňový priečinok pre Content . Content je zložka, v ktorej sú naše herné dáta.

Okrem konštruktoru tu máme aj metódu Initialize ( ). To môže byť mätúce. Microsoft vlastne nijako nehovorí, kedy použiť k inicializácii konštruktor a kedy metódu Initialize.

Rozdiel je v tom, že konštruktor sa volá hneď pri vytvorení hry (alebo herné komponenty) a mal by nastaviť hru alebo komponent tak, aby bola funkčná.

Metóda Inicialize( ) potom slúži na načítanie dát, ktoré nie sú obsah (Content, teda nie sprity, zvuky alebo hudba), môžu to byť napríklad nejaké súbory máp. Tiež tu prevedieme všetku inicializáciu hry, vytvorenie potrebných objektov, nastavenie premenných na predvolené hodnoty a podobne.

V initialize máme vložený riadok kódu :

*base.Initialize( ) ;*

Ten sa stará o spustenie metódy Initialize ( ) na všetkých komponentoch hry. Podobný riadok nájdeme aj v ďalších metódach triedy .

## Load Content

Ďalej máme v triede metódu LoadContent ( ), do tej patrí načítanie obsahu, teda sprites, zvukov a hudby. Vidíme, že sa tu vytvára aj spriteBatch. Na koniec metódy môžeme písať logiku, ktorú potrebujeme spustiť až po načítaní obsahu. Napr. nemôžeme v Initialize ( ) zistiť výšku fontu, ktorý ešte nie je načítaný, preto by kód patril do LoadContent.

UnloadContent sa volá po skončení hry, pretože my budeme vždy používať Content (čo je vlastnosť hry, kde je inštancia Contentmanager ), ktorá si uvoľnenie zdrojov rieši sama, nie je pre nás metóda dôležitá.

Nasledujú 2 najdôležitejšie metódy :

## Update

Update ( ) obsahuje real - time logiku, teda všetko, čo sa spracováva v reálnom čase. Väčšinou je to klávesnica, prípadne myš alebo iné ovládače, potom kolízie a pohyb objektov v hre. Všeobecne sem patria reakcie na nejakej udalosti a posun objektov, prípadne ich animácie. Metóda sa vykonáva 60x za sekundu. Ak posunieme postavu v tejto metóde o 1, bude chodiť rovnako rýchlo aj na inak rýchlom počítači. Iný interval môžeme nastaviť ako desatinné číslo menšie ako 1 pomocou vlastnosti hry TargetElapsedTime, ale to nebudeme potrebovať. Dôležité je vedieť, že XNA za nás bude samo robiť optimalizáciu, aj keď počítač nebude hru stíhať a to tak, že bude vynechávať vykresľovanie.

V parametri metódy, dostaneme inštanciu GameTime. Tu je uložený herný čas. Nájdeme na ňom 2 užitočné vlastnosti:

ElapsedGameTime - TimeSpan s časom ubehnutým od posledného update,

TotalGameTime - TimeSpan s celkovým časom behu hry.

Vďaka týmto hodnotám môžeme v Update pracovať s reálnym časom, niečo treba urobiť každú sekundu, minútu a podobne.

Ďalšia vlastnosť na gametime je IsRunningSlowly, tá je true, ak hra "seká" a XNA vynecháva vykresľovanie. Môžeme tak na túto situáciu reagovať.

V metóde je už doplnená jedna reakcia na tlačidlo Späť na gamepade, ktorá ukončí hru. Je to dôležitý kód preto, aby išla hra ukončiť na Xboxe, ak by sme ju tam chceli nahrať. Okrem iného vidíme, že ukončenie hry sa vykoná pomocou metódy Exit ( ).

## Draw

Draw ( ) sa stará o vykreslenie hry (ako už názov napovedá). Vykresľovanie prebieha volaním metód na inštanciu SpriteBatch. Grafické zariadenie musíme pred každou snímkou ​​(frame) vymazať, aby na ňom nezostali vykreslené objekty z minula. Toho je docielené metódou Clear a modrou farbou (na farbe vlastne nezáleží, ak bude mať hra pozadie). Stretávame sa tu so štruktúrou Color , tá v XNA slúži k ukladaniu farieb. Má na sebe niekoľko statických metód, ktoré navrátia jej inštanciu nastavenú na určitú farbu.

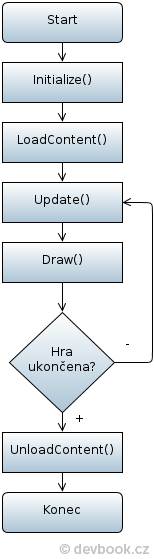
## Herná slučka

Je dôležité vedieť, ako vo vnútri hra pracuje. Herné metódy sa volajú v tzv. hernej slučke a to v tomto poradí :

* Initialize
* LoadContent
* Update
* Draw
* UnloadContent

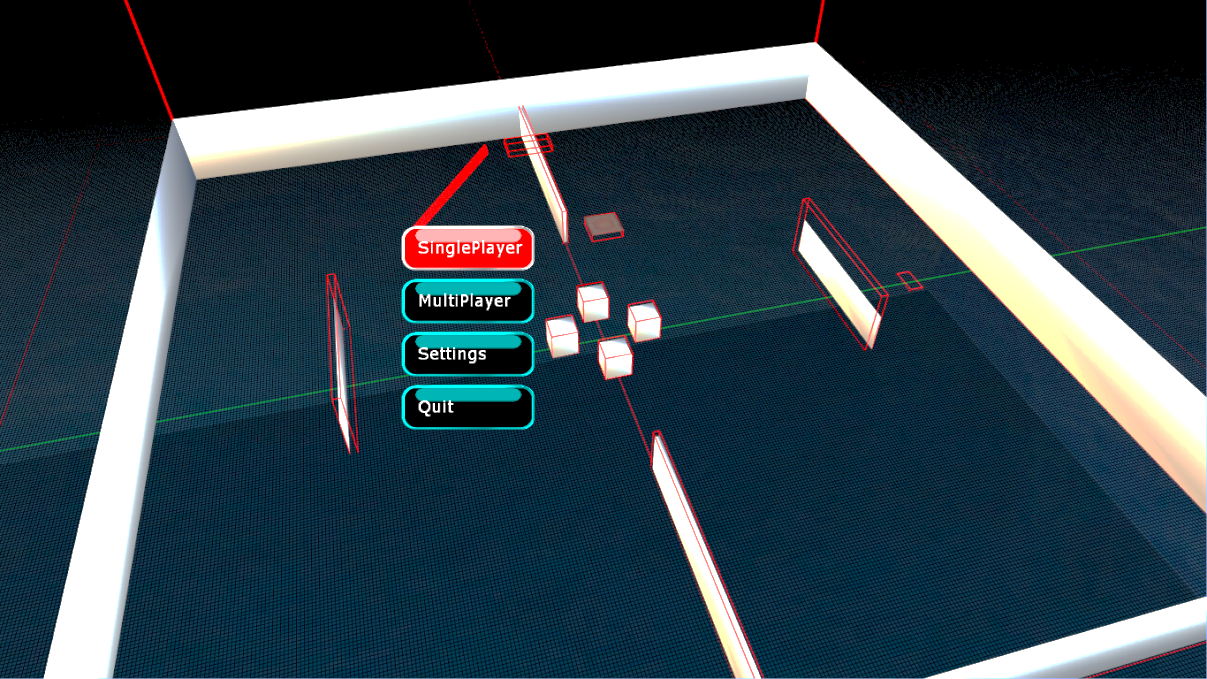
Update a Draw sa stále opakujú, až kým nie je hra ukončená. Cyklus znázornený pomocou vývojového diagramu by vyzeral tak, ako to môžeme vidieť na obrázku Obr. 1:

Obr.



# Herné Menu

Ako prvé, keď zapneme hru, sa zobrazí hlavné menu, na ktorom je vidno 4 tlačidlá

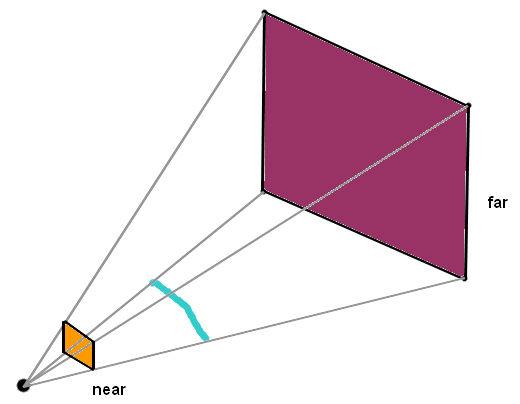
* Singleplayer - pre spustenie singleplayerovej hry
* Mutiplayer - pre spustenie multiplayerovej časti hry
* Settings – ktoré zatiaľ nie je obslúžené
* Quit- pre ukončenie hry

Obr.

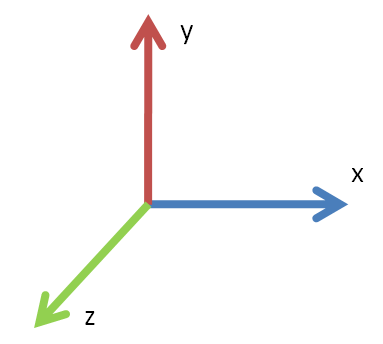
# Kamera

Najprv sme si vyrobili triedu Kamera, v ktorej sme vytvorili základy kamery. Ďalej sme si vytvorili triedu TargetKamera, ktorá bola len statická a pozerala sa z určitej pozície na určitý bod. Ďalej sme kameru museli upravovať a na toto sme si vytvorili triedu FreeKamera – je to kamera, ktorá sa môže pohybovať všetkými smermi, teda po osách X, Z ale aj Y, podľa natočenia kamery.

Obr.



.**Obr. 4**



Toto riešenie sme mohli používať pri vývoji, ale pre Hráčov by to nebolo najlepšie riešenie. Preto sme si vytvorili triedu FPSKamera, v ktorej sa už hráč mohol pohybovať po osách X a Z ale Y ostávalo stále rovnaké. Aby sa hráč vedel pohybovať aj po Y ose (skákať/padať), museli sme vymyslieť špeciálny algoritmus.

Algoritmus pre skákanie:

1. Ray vyska = new Ray(Pozice, new Vector3(0,-1,0));
2. testray = vyska;
3. if (jumping)
4. {
5. float minimum=-100000;
6. foreach (BoundingBox Box in Parent.engine.stenyF)
7. {
8. if (vyska.Intersects(Box) != null)
9. if (minimum < PoziceY - (float)vyska.Intersects(Box)+70)
10. minimum = PoziceY - (float)vyska.Intersects(Box)+70;
11. }
12. PoziceY -= jumpspeed;
13. jumpspeed += 1;
14. if (PoziceY <= -10000)
15. Pozice = new Vector3 (0,10000,0);
16. else
17. startY = minimum;
18. if (PoziceY <= startY){
19. PoziceY = startY;
20. jumping = false;
21. jumpspeed = 0;
22. }
23. }
24. else{
25. if (Parent.Engine.Input.StisknutaKlavesa(Keys.Space)){
26. jumping = true;
27. jumpspeed = -25;
28. startY = PoziceY;
29. }
30. }
31. Parent.engine.rychlostpadania = jumpspeed;
32. kolidujeY = false;
33. bool d = false; float min=1000000;
34. if ( min < 70 && min > 0)
35. {
36. PoziceY += min;d = true;
37. }

A pre padanie:

1. if(!d)
2. if (!jumping) {
3. float minimum = -100000;
4. foreach (BoundingBox Box in Parent.engine.stenyF)
5. {
6. if (vyska.Intersects(Box) != null)
7. if (minimum < PoziceY - (float)vyska.Intersects(Box) + 70)
8. minimum = PoziceY - (float)vyska.Intersects(Box) +70;
9. } PoziceY -= jumpspeed;
10. if (PoziceY <= -10000)
11. {
12. PoziceY = 10000;
13. PoziceX = 0; PoziceZ = 0;
14. }
15. else
16. startY = minimum;
17. if (PoziceY <= startY)
18. {
19. PoziceY = startY;
20. jumpspeed = 0;
21. }
22. else
23. {
24. jumpspeed += 1;
25. }
26. }

Tento algoritmus využíva triedu Ray, ktorá pomocou metódy. Intersects(BoundingBox Box) zistí, v akej vzdialenosti koliduje (pretína) s objektom Bounding Box, ktorým sú obalené všetky steny a podlahy.

# Kolízie

Na kolízie vieme využiť v XNA už  predpripravené triedy:

* BoundingBox
* BoundingSphere
* Ray

Na testovanie, či daný objekt koliduje s iným objektom, použijeme metódu Intersect();

# SkyBox

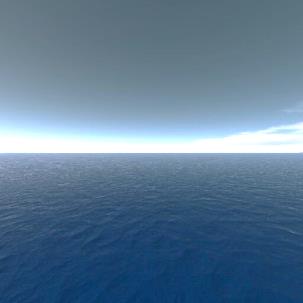
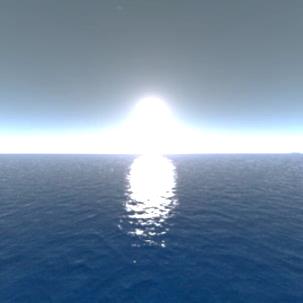
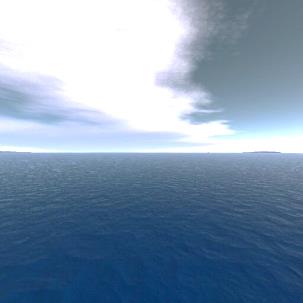
Je to prostredie (svet) okolo hry. Skladá sa z 6 častí:

* Top (Obr.5)
* Front(Obr.6)
* Right(Obr.7)
* Back(Obr.8)
* Left(Obr.9)
* Bottom(Obr.10)

Obr.5



**Obr.6 Obr.7 Obr.8 Obr.9**



**Obr.10**



# SinglePlayer vs Multiplayer

Hra sa rozdeľuje na 2 časti:

* SinglePlayer
* Multiplayer

Ako nepriateľov sme použili model od Microsoftu- „dude“.

## SinglePlayer

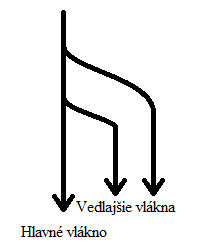
V singleplayerovom móde máme možnosť bojovať proti počítačom riadenými nepriateľmi. Pohyby AI nepriateľov sú náhodné generované.

## MultiPlayer

V multiplayerovom móde máme možnosť hrať proti skutočným ľuďom.

Na tvorbu sme použili triedu Socket, ktorá nám umožňuje komunikáciu cez server. Pri tvorbe servera sme mali problém s tým, že kým server čakal na pripojenie ďalších hráčov, nebolo možné, aby sa dáta odosielali už pripojeným hráčom. Tento problém sme vyriešili pomocou multithreadu. Ten sa skladá z viacerých vlákien (obr. 11).

Obr. 11

A na čo sa vlákna požívajú? Jedno vlákno sa staralo o pripojenie hráčov a jedno sa staralo o prijímanie a odosielanie dát hráčom.

Podobné riešenie sme museli dať aj do samotnej hry, aby sme mali vždy aktuálne dáta, a keď sa buffer socketa zaplní, nezasekne sa hra. Hlavné vlákno sa stará o hru, ďalšie vlákno sa stará o odosielanie dát na server a ďalšie vlákno sa stará o prijímanie dát zo servera.

# Ovládanie hry

Hru vieme ovládať pomocou klávesnice a pomocou myšky.

Ovládanie:

W – pohyb dopredu , výber položky menu

S – pohyb dozadu, výber položky menu

A – otočenie sa doľava

D –otočenie sa doprava

Q – pohyb do boku - doľava

E - pohyb do boku - doprava

Space – skok

Enter – výber položky menu

ESC- v hlavnom menu- koniec, v hre- zobrazenie ingame menu

Aby sme videli kolízne krabice, ktoré obaľujú steny, stlačíme tlačidlo F7.

Na vývojárske potreby, aby sme videli herné parametre ( pozícia XYZ, natočenie kamery Pitch a Yaw, atď.), stlačíme F1.

V hlavnom menu máme možnosť spustenia rovnakého módu (singleplayer / multiplayer) ako bolo spustené posledný krát pomocou tlačidla F3.

# Minimálne požiadavky

Hardvérové:

* podporovanú grafickú kartu s aktuálnym ovládačom
* Zvuková karta: DirectX 9.0c Compatible

Softvérové – hra vyžaduje, aby ste na počítači mali nainštalované:

* Microsoft XNA Framework Redistributable 4.0
* .NET Framework 4.0
* Microsoft Visual Studio C# 2010 Express

# Záver

Pri písaní tejto práce sme si čím ďalej, tím viac uvedomovali, aký náročný je vývoj a tvorba hier. Pri tom sme narazili aj na veľmi veľa problémov, ktoré sme sa pokúšali vyriešiť rôznymi cestami, žiaľ, nie všetky sa nám podarilo vyriešiť, kvôli nedostatočným vývojovým podmienkam.

V budúcnosti by som chcel s hrou ďalej pracovať a vylepšiť ju. Chcel by som dokončiť plnohodnotný multiplayer, na ktorý by sa mohlo pripojiť viac ľudí, čo sa mi zatiaľ nepodarilo, kvôli nedostatočnému synchronizovaniu dát. Ďalej by som chcel dokončiť AI(umelú inteligenciu), ktorého pohyby by už neboli náhodné, ale by boli dôkladne prepočítané a vyhodnotené a aby aj v singleplayer-ovom móde sme mali súperov, ktorý by strieľali aj po nás, nie len my po nich.

Taktiež by som chcel, aby sa modeli nepriateľov pohybovali, a neboli by len statické. Poslednú vec, čo by som ešte chcel vylepšiť je územie. Teraz ho máme len veľmi jednoduché, v ktorom nie je dostatok možností na pohyb.

# Resumé

When writing this work, we start to understand how hard it is to create a computer game. While creating the game, we faced a great many problems that we tried to solve in various ways, unfortunately, not all we managed to solve, due to lack of developmental conditions. The Game was made in the programing language C# and we used the Microsoft XNA v4.0 framework. The Game is both single and multi-player. In the singleplayer part of the game, the player plays against the enemies which are controlled by computer. As the name suggests multiplayer game is for more players. In the multiplayer part of the game two players can face each other in a duel. We need a server for the multiplayer to work properly. For the server data transport we use one of the C# classes – Socket.

# Zoznam bibliografických odkazov

1. Vodáček, 2012, C# - *Základy 3D grafiky a tvorba enginu*

David Čápka, 2013 - *Od nuly k tetrisu v XNA game studio*  
<http://www.devbook.cz/c-sharp-xna-game-studio-tutorialy-na-tvorbu-her>,

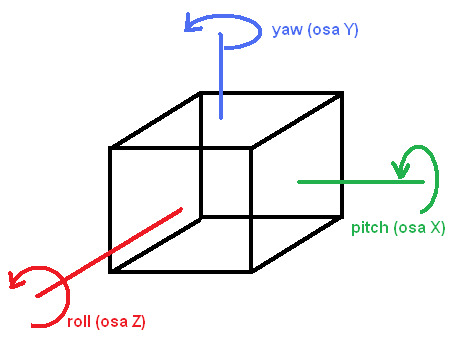
1. Herceg, Tomáš, 2008, *Vykreslujeme 3d model v xna*  
   <http://cdn.dotnetportal.cz/files/WindowsLiveWriter-Vykreslujeme3Dmodel_AEE5-image_7d623214-90eb-4735-95b7-f255425d854d.png>
2. Red Sorceress, <http://www.redsorceress.com/skybox.html>

# Prílohy

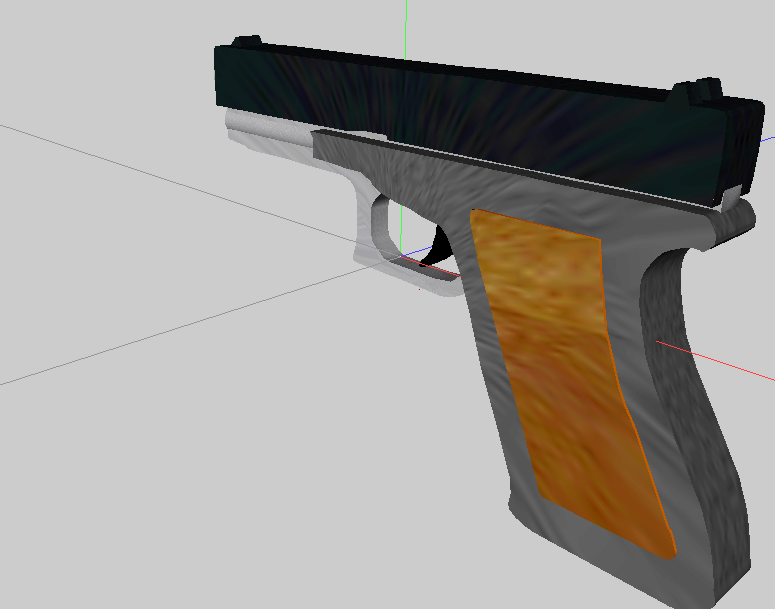
Obr.12 - 3D model “dude” od spoločnosti Microsoft



Obr.13 – rotácia



Obr.14 – Zbraň použitá v hre



Obr.15 – Ukážka z hry

