Space Invaders

Dokumentácia k zápočtovému programu

Matej Bujňák

Užívateľská dokumentácia

Témou môjho zápočtového programu je jednoduchá implementácia klasickej GUI singleplayer hry Space Invaders prevedená v jazyku C# s použitím Windows Presentation Foundation (WPF). Hra ponúka jednoduchú grafiku, nastavenia keybindov, voľbu ovládania myšou/klávesnicou a systém vĺn nepriateľov.

Nepriatelia sú rozmiestnení do rôznych tvarov na základe textových súborov v koreňovom adresári a strieľajú po hráčovi, zatiaľ, čo úloha hráča je vyhýbať sa im, ich projektilom v snahe zostreliť ich, aby postúpil do ďalšej vlny.

Užívateľská dokumentácia obsahuje užívateľskú časť pre bežného užívateľa. Popisuje priebeh hry, UI elementy, interakciu s hrou a ďalšie informácie.

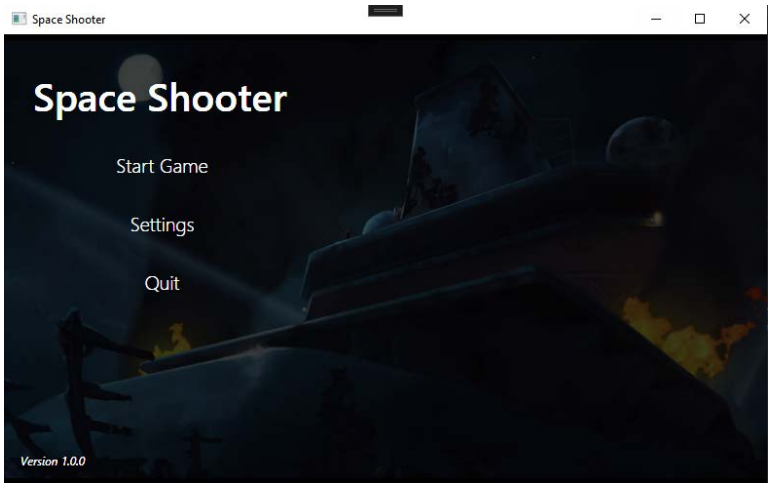
Pravidlá hry:

Ide o singleplayer hru, teda hru pre jedného hráča. Hráč ovláda vesmírnu loď so schopnosťou hýbať sa v štyroch smeroch (vpravo, vľavo, hore, dole) a má schopnosť strieľať. Jeho úlohou je zabíjať nepriateľov, ktorí strieľajú naspäť po ňom. Každý nepriateľ má inú odolnosť a hráč za každého dostane iný počet bodov. Keď je hráč trafený nepriateľským projektilom, stráca život. Hra končí tým, že buď hráč zomrie (stratí všetky životy), alebo zabije všetkých nepriateľov v každej vlne. Ak sa hráč zrazí s nepriateľom, nebudú mu udelené žiadne body a bez ohľadu na odolnosť nepriateľa, nepriateľ zomrie.

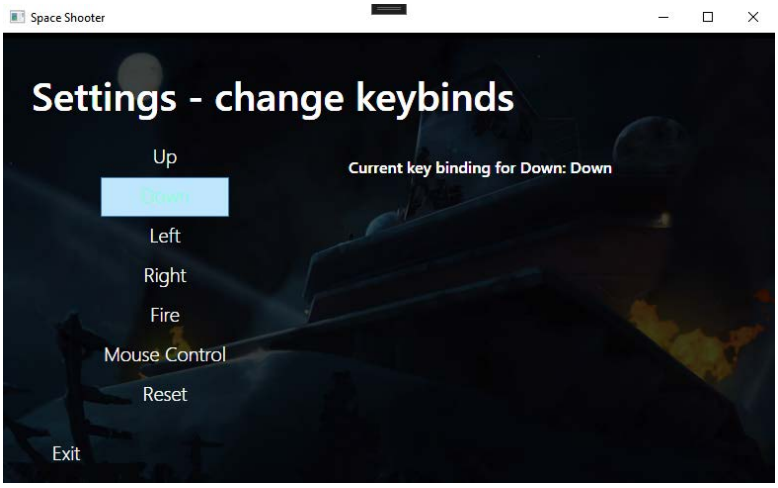
Priebeh hry:

Po spustení sa používateľ ocitne v hlavnom menu, ktoré obsahuje 3 možnosti:

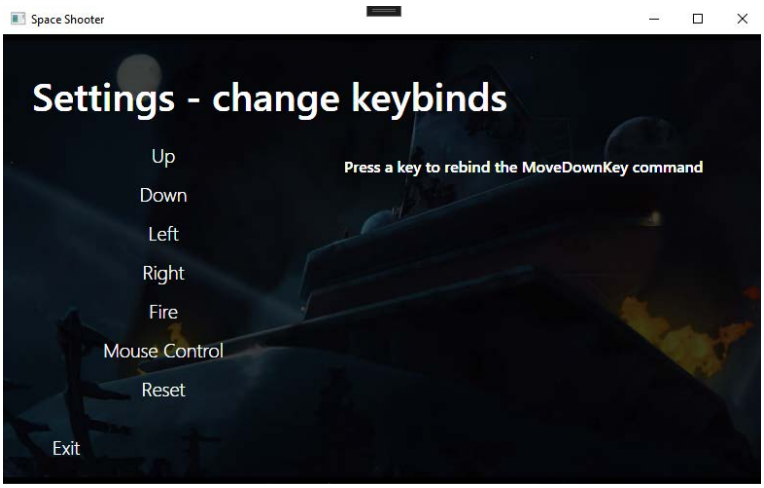
- *Start Game* – zapne hru s predvolenými nastaveniami
- *Settings* – otvorí nastavenia a dáva používateľovi možnosť meniť nastavenia keybindov/možnosť používať myš na ovládanie
- *Quit* – Ukončí aplikáciu



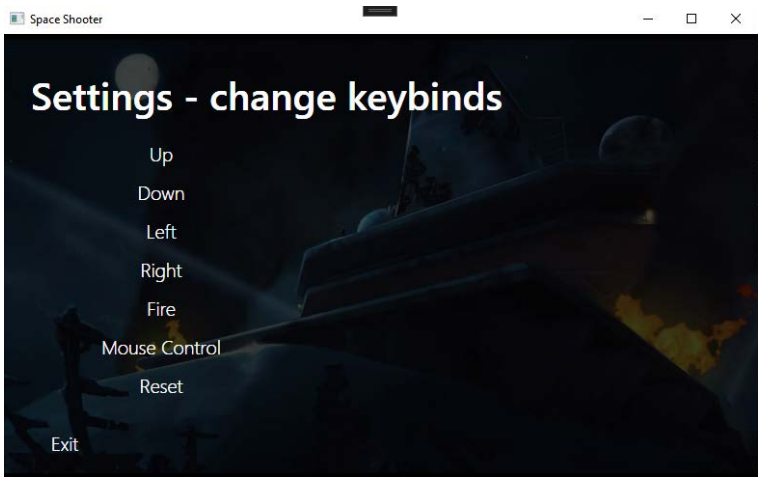
Zvolením možnosti *Settings* sa používateľ ocitne v nastaveniach, kde má na výber možnosti, ktoré ovládacie prvky môže na klávesnici premapovať na iné klávesy:



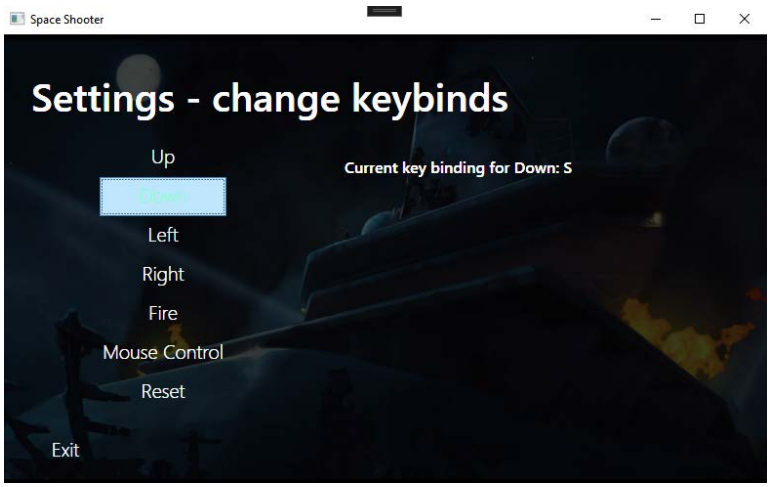
Možnosti *Up*, *Down*, *Left*, *Right* a *Fire* priamo korešpondujú ovládacím prvkom vesmírnej lode hráča. Pri presunutí kurzoru myši na ľubovoľné tlačidlo používateľ uvidí, ktorá klávesa je aktuálne priradená danej akcií. Po stlačení tlačidla vyzve *Prompt*(text napravo) na stlačenie ľubovoľnej klávesy, ktorá bude následne priradená danej akcii:



Po stlačení klávesy, ktorú by používateľ chcel priradiť danej akcii *Prompt* zmizne:

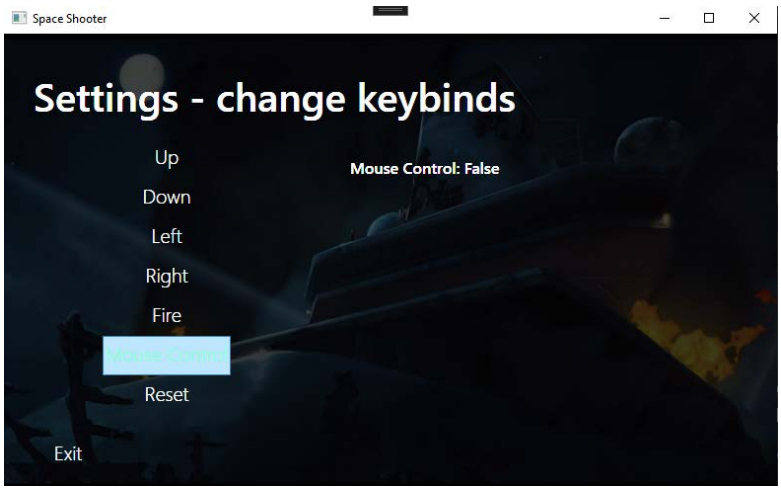


Po ďalšom prejdení myšou po danom tlačidle používateľovi *Prompt* zobrazí zmenenú klávesu:



V tejto ukážke som priradil pohybu hráča dole klávesu 'S' namiesto prednastavenej klávesy 'Down' (šípka dole).

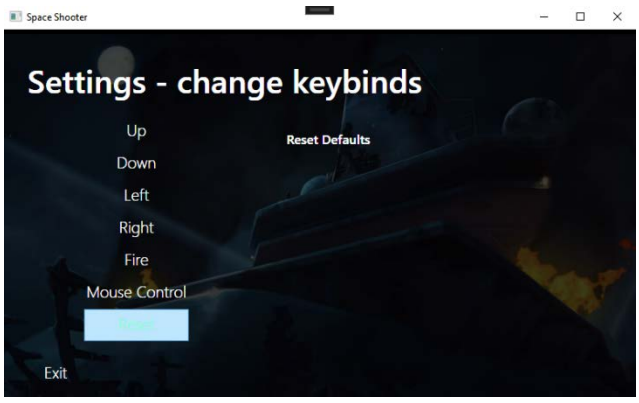
Tlačidlo Mouse Control je len prepínač. Ukazuje svoj aktuálny stav (True/False):



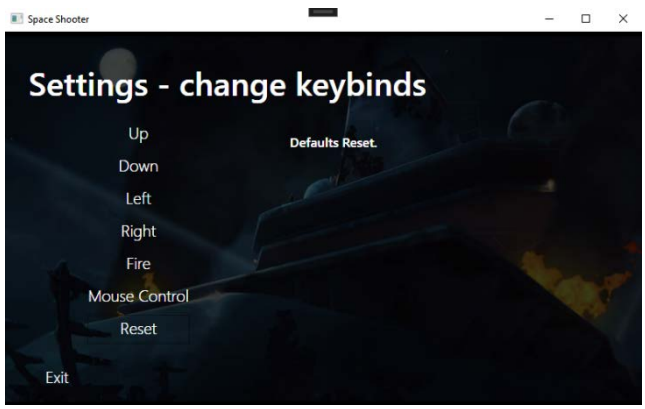
Ak používateľ nastaví tlačidlo do polohy True, tak vesmírna loď bude ovládaná kurzorom myši a streľba bude ovládaná LMB (ľavým tlačidlom myši). V prípade, že tlačidlo bude v polohe False, tak hra bude reagovať na klávesové ovládacie prvky.

Posledným tlačidlom je tlačidlo Reset, ktorého úlohou je obnoviť predvolené nastavenia:

Pred stlačením



Po stlačení



Predvolené nastavenia:

Up (*Pohyb hore*) .....W

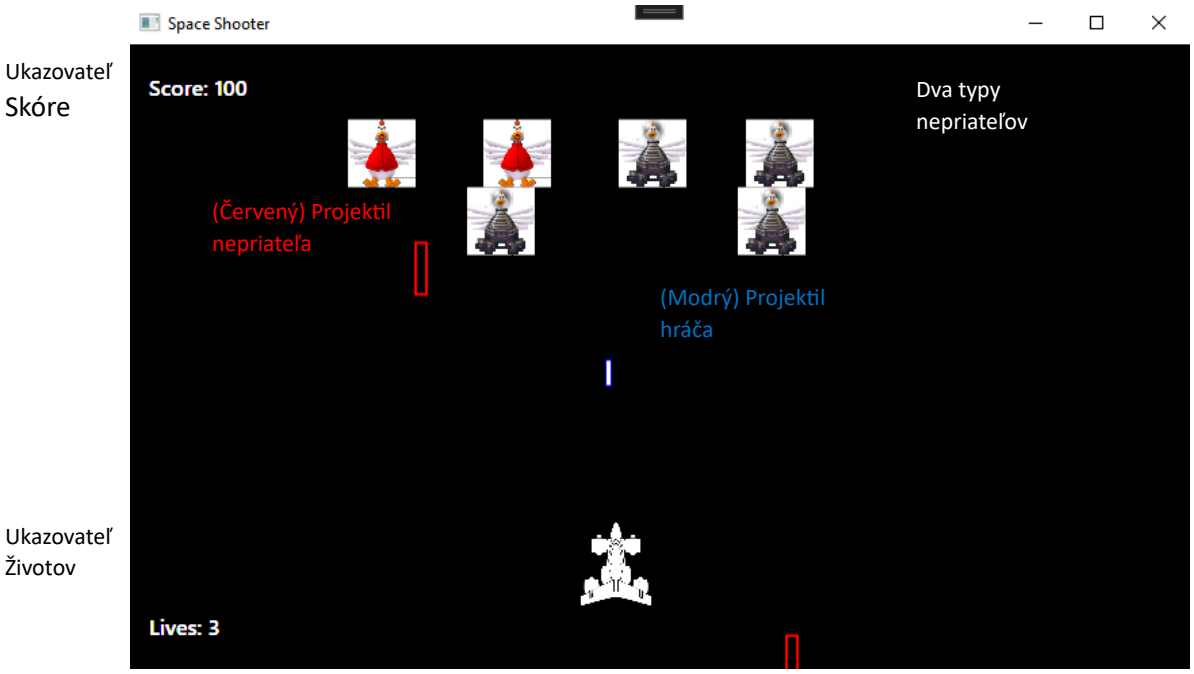
Left (*Pohyb vľavo*) .....A

Down (*Pohyb dole*) .....S

Right (*Pohyb vpravo*) .....D

Fire (*Streľba*) .....Space

Zvolením možnosti *Start Game* hráč spustí hru. Na hernej obrazovke je možné vidieť viacero vecí:



Typy nepriateľov:



Život:	2	3	4
Hodnota v bodoch:	100	300	400

Programátorská dokumentácia

Základné informácie:

Program je vytvorený vo Windows WPF. Pozostáva z jedného okna (*Window1.xaml*) a zo štyroch UserControlov – *MainMenuWindow.xaml*, *MainWindow.xaml*, *PauseMenu.xaml* a *Settings.xaml.* Pri Spustení programu sa vytvorí *Window1*, ktorého obsah bude vyplnený *MainMenuWindow* UserControlom. Ten je prepojený s ostatnými UserControlmi prostredníctvom Buttonov, ktoré sú definované v .xaml súboroch a implementované v .xaml.cs súboroch.

Prehľad metód v jednotlivých súboroch:

MainMenuWindow.xaml.cs:

Obsahuje Event Handling metódy pre tlačítka definované v .xaml súbore

StartGame() – spustí hru tým, že vytvorí objekt mainWindow, ktorého obsahom vyplní aktuálne okno a spustí hru funkciou StartGame()

ShowSettings() – otvorí nastavenia, tým, že obsah aktuálneho okna vyplní obsahom vopred vytvoreného objektu Settings()

Quit() – ukončí aplikáciu

MainWindow.xaml.cs:

Obsahuje hlavnú hernú logiku, hitboxy, triedy MainWindow, Enemy a GameConfiguration, pohyb hráča a nepriateľov, pohyb projektilov, kolízie, čítanie a parsovanie súborov waves.txt a config.txt, loading textúr pre hráča a nepriateľov, funkcie na resetovanie hry, pause menu eventy, odstraňovanie nepotrebných objektov, animáciu výbuchu, a podobne.

Enemy – je to trieda reprezentujúca parametre nepriateľa, obsahuje parametre:

Rectangle – vizuálna reprezentácia nepriateľa pomocou Rectangle objektu, ktorý je neskôr vyplnený správnou textúrou,

Health – int hodnota udávajúca počet zásahov potrebných na zostrelenie nepriateľa,

PointValue – int hodnota udávajúca počet bodov, ktoré hráč dostane pri zostrelení nepriateľa,

ProjectileColor – Brush udávajúci farbu projektilu nepriateľa

Objekty tejto triedy vytvára metóda *SpawnEnemy()* z triedy **MainWindow**, ktorá berie ako parameter typ nepriateľa, a súradnice, kam ho má umiestniť. Podľa typu nepriateľa (definovaného v enum EnemyType) vytvorí Enemy objekt s parametrami korešpondujúcimi k typu nepriateľa.

GameConfiguration – trieda, ktorá udržiava keybindy. Pri spustení programu sa vytvorí objekt gameConfiguration, ktorý následne vyplní metóda *LoadKeybindsFromFile()*, ktorá číta obsah súboru *config.txt*. V prípade, že súbor je nesprávne formátovaný, alebo chýba, tak objekt gameConfiguration táto metóda vyplní default hodnotami (konštantami) definovanými v triede *MainWindow*.

MainWindow - trieda ktorá má na starosti prakticky celú hernú logiku od ovládania, cez hitboxy, až po čítanie wave patternov z textových súborov. Pri spustení trieda definuje konštanty, potrebné premenné a vytvorí všetky potrebné Listy, Brushes, Timery, PauseMenu a MainMenuWindow objekty, načíta snímky animácie výbuchu a priradí button click eventom z pauseMenu metódy.

Každý Timer má svoju funkciu, ktorá sa vykoná každých niekoľko milisekúnd (500, 50, 5).

enemyShootingTimer – má za úlohu každých 500ms vybrať náhodného žijúceho nepriateľa pomocou metódy *EnemyShootingTimer_Tick()*, ktorý vystrelí.

animationTimer – pomocou metódy *AnimationTick()* prehráva animáciu výbuchu hráča

gameTimer – pomocou metódy *GameLoop()* aktualizuje celú grafiku v okne a rieši hernú logiku

Metóda GameLoop() – vykoná sa každým Tickom gameTimeru (každých 5ms). Pomocou metódy *updateUI()* aktualizuje skóre a životy. Na základe toho, či je boolean *MouseControl* z *gameConfiguration* nastavený na True alebo False použije jednu z handlingových metód pre pohyb – *MovePlayerMouse()* alebo *MovePlayer()*.

MovePlayerMouse() umiestňuje hráča tam, kde ukazuje kurzor myši, zatiaľ čo *MovePlayer()* sa riadi booleanmi *goLeft, goRight, goUp, goDown* na základe *metód KeyIsUp()* a *KeyIsDown()*.

Ďalej *for cyklom* prechádza všetky existujúce objekty typu *Rectangle* a podľa Tagu rieši, čo s nimi. Ak ide o projektil hráča:

pomocou metódy *solveCollisions()* jednak rieši pohyb projektilu po obrazovke, pomocou metódy *removeProjectilesOutOfTheWindow()* zistí, či projektil je vyššie ako definovaná hranica, a ak je, tak ho odstráni. Potom vlastným *for cyklom* prechádza všetkých nepriateľov a vytvorením *hitboxov* pomocou *Rect* objektov zisťuje, či došlo ku kolízii. Ak áno, tak rieši kolíznu logiku.

Ak ide o nepriateľa (enemy) :

pomocou metódy *moveEnemy()* posunie nepriateľa a následne kontroluje kolíziu s hráčom pomocou metódy *playerEnemyCollision()*

Podobne to vyzerá, ak ide o nepriateľský projektil (enemyProjectile). Pomocou *moveProjectile()* ho posunie na novú pozíciu a pomocou *playerEnemyCollision()* skontroluje kolíziu s hráčom.

Ďalej sa v každom Ticku GameLoopu odstraňujú nepotrebné objekty pomocou *removeUnusedItems()* (tie sú tam pridané, keď projektil vyjde mimo okna/keď zomrie nepriateľ...) a skontroluje sa víťazná podmienka pomocou *checkGameOver().* Ak sú všetci nepriateľia mŕtvi a waveNumber<=maxWaveNumber (teda aktuálna wave nie je posledná), tak sa gameTimer pozastaví, načíta sa nový wavePattern, zvýši sa waveNumber, gameTimer sa znovu spustí a začne ďalšia wave.

Ďalej trieda obsahuje rôzne metódy na spawnovanie objektov ako *enemyProjectileSpawner()* – *EnemyShootingTimer_Tick()* , *explode()* – *playerDeath() a playerEnemyCollision(), Shoot()* - *MouseButtonDown()* , ktoré sú nevyhnutné pre správnu funkčnosť hry.

Metódy *LoadWavePatterns()*, *ParseEnemyType()*, *StartNextWave()*, *SpawnEnemiesFromWavePattern()* a *spawnEnemy()* slúžia na čítanie textových súborov wave{1-3}.txt a rozmiestňovanie nepriateľov do okna podľa nich.

Metódy *SwitchToMainMenu(), PauseGame()*, *ResumeGame()* sú priamo naviazané na tlačidlá z Pause Menu.

Metódy *showGameOver()* – zastaví timery, a ukáže hráčovi text o tom, či vyhral alebo prehral. *ResetGame()* – resetuje všetky parametre na ich pôvodné hodnoty, poodstraňuje všetky nepotrebné objekty a začne hru znova metódou *StartGame()*.

Settings.xaml.cs:

Poskytuje metódy pre zmenu keybindov, na čítanie a zápis do konfiguračného súboru, na čítanie stlačenej klávesy, na vstup kurzora do tlačidla, na obnovenie default nastavení a na prepínanie naspäť do *MainMenuWindow*.

Rebind() – nastaví aktuálny control na rebind a zobrazí ho v *Prompte*

UserControl_KeyDown() – má na starosti rebinding eventy, keď má *Settings Focus*. *Sender*-om je aktuálny *UserControl*, ale *e* je objekt triedy *KeyEventArgs*, ktorý obsahuje informáciu o tom, ktorá klávesa bola stlačená. Podľa toho, ako bola nastavená *currentControlToRebind* nasleduje switch a na základe toho bude nejaký control v *gameConfig* prepísaný. Pred ďalším použitím sa *currentControlToRebind* zresetuje a aktualizuje sa konfiguračný súbor použitím metódy *updateConfigFile()*.

updateConfigFile() – v správnom formáte prepíše konfiguračný súbor *config.txt* s novými parametrami.

Button_MouseEnter() – Handling pre Event, kedy kurzor myši vojde do tlačidla. Podľa toho, do ktorého tlačidla vojde, *switch* statement zobrazí v *Prompte* aktuálny keybind pre daný control.

UpRebind(), DownRebind(), LeftRebind(), RightRebind(), a *FireRebind()* – spravujú ButtonClick eventy tak, že zavolajú metódu *Rebind()* pre daný control.

MouseControlToggle() – prehadzuje bool hodnotu *mouseControl* z *gameConfiguration* a zobrazuje ju v *Prompte.*

ResetDefaults() – Resetuje nastavenia tým, že ich nahradí default definovanými v *MainWindow*

SwitchToMenu() – prepne naspäť do menu vytvorením *MainMenuWindow* objektu a nahradením contentu zaň

PauseMenu.xaml.cs:

Reprezentácia Pause Menu počas hry. Obsahuje 3 tlačidlá definované v .xaml súbore a tu je ich implementácia. Rovnako ako MainMenuWindow, obsahuje Event Handling metódy, ktoré spravujú eventy stlačených tlačidiel.

ResumeButton_Click() – Keď je stlačené tlačidlo *Resume* v Pause Menu, vyvolá to ResumeClicked Event v MainWindow a teda metódu *ResumeGame()*

RestartButton_Click() – Keď je stlačené tlačidlo *Restart* v Pause Menu, vyvolá to RestartClicked Event v MainWindow a teda metódu *ResetGame()*

ExitButton_Click() – Keď je stlačené tlačidlo *Exit* v Pause Menu, vyvolá to ExitClicked Event v MainWindow a teda metódu *SwitchToMainMenu()*