Tetris – Dokumentace

*Zápočtový program, Tetris, je zábavný program, který lze využít pro zabití času hraním legendárních her i pro edukativní cíle. Program obsahuje klasickou hru tetris, s typickým bodovým systémem a pravidly, která jsou drtivé většině lidí na planetě Zemi známy. Mimo to program obsahuje 2 různé umělé inteligence, které (za vhodných podmínek) dokážou hrát tetris i hodiny. Nejenom herní pamětníci zde najdou další hru, kterou nejspíš ve svých mladých létech pařili na gameboyích.*

# Programátorská sekce

## Část I – Hra Tetris

Kapitola 1 – hrací deska, místo pro radost i rozruch

Hrací deska pro tetris a obě umělé inteligence v daném projektu je reprezentována pomocí objektu třídy GameBoard, který je v hlavní třídě Form1.cs uložen pod názvem gb. Při zapnutí (nebo i přepnutí) kteréhokoliv režimu se vždy zinicializuje nová hrací deska – gb = new GameBoard(); –. Objekt této třídy obsahuje dvojrozměrné pole public char[,] Board, které je v konstruktoru defaultně naplněno nulovým charem ‘\0‘, což je chápáno jako prázdné pole, místo, kam se může TetroBlock posunout. Hrací deska má rozměry 10x20, ale zobrazováno je jenom dolních 18 řad. Horní 2 sice nejsou zobrazovány, ale i přesto může hráč (například při rotaci) do nich zavítat. Tato třída kromě hracího pole také obsahuje celočíselné hodnoty level, lines a score. Nejpodstatnější z této trojice čísel je level, tato hodnota upravuje rychlost hry. Dochází ke zrychlení hry až do 10. levelu (kdy je rychlost hry 140ms), pak už je zrychlení nulové.

Třída GameBoard je mimo jiné používána pro generaci TetroBlocků. Generace není přímo náhodná. Existuje 7 druhů TetroBlocků, se kterými můžeme hrát tetris. Generaci si můžeme představit následovně. Do měšce vložíme všech 7 existujících TetroBlocků a postupně je vytahujeme po jednom. V okamžiku, kdy je měšec prázdny, navrátíme všechny TetroBlocky zpět a začínáme od znova. Tím pádem je záruka, že každých 13 generací (v nejhorším případě) dostanu chtěný TetroBlock. Tuto úlohu řeší statická metoda static public Shape GeneratePiece(), která ukládá potřebné informace do statických proměnných static int numOfPieces a static bool[] piecesDistribution. Návratová hodnota této funkce je Shape, což je abstraktní mateřská třída všech 7 různých TetroBlocků, proto můžeme jednoduše dosadit a přetypovat proměnnou Shape na potřebný objekt. Další podstatné metody této třídy jsou:

* public bool AddToBoard(Shape shp)
* public int[] FindFullLines()/static public int[] FindFullLines(ref char[,] deska)
* static public void MoveMap(ref char[,] deska, int[] lines)
* static public void ClearLines(ref GameBoard gb, int[] lines)

První zmíněná metoda vrátí true, pokud se nám podaří vložit aktivní hrací TetroBlock do hrací desky, tím pádem patří ke stěžejním metodám, která nám prozradí, kdy je třeba hru ukončit, protože už není místo pro další TetroBlock. Jedná se o klasický cyklus přes pole int[,] Pozice, který otestuje, zda každá pozice v poli char[,] Board má nulový char jako hodnotu.

kapitola 2 – malujeme kostičky

Formulářová aplikace obsahuje celkově 20 prvků: 3x PictureBox, 5x Button, 5x Timer, 6x Label a 1x CheckBox. Všech 6 labelů složí pro zobrazování aktuálních údajů běžící hry. CheckBox se používá pro zapínání a vypínání Tetris Theme Song. Funkce každého tlačítka je zřejmá z textu, který je napsán na jeho tlačítku. Při přepnutí do kteréhokoliv režimu se nejprve vypnout všechny timery, vynulují se globální proměnné pro chod programu, vytvoří se nové objekty s „čistým štítem“, nastaví se správně hodnoty score a vygenerují se nové TetroBlocky, které jsou uloženy v proměnných Shape activePiece a Shape nextPiece. Jak už název napovídá, tak tety proměnné definují figurku, se kterou hráč (popř. umělá inteligence) aktuálně hraje a manipuluje na hracím poli. Figurka uchovává svoji barvu a svoje 4 pozice, na kterých se „virtuálně“ nachází v hracím poli. Je zde použito slovo virtuálně, protože dokud je figurka aktivní a je s ní možnost zacházet a manipulovat, tak není zaznamenána v hracím poli. K této změně dojde po zavolání metody třídy GameBoard: public bool AddToBoard(Shape shp). Nachází se zde 5 timerů, 2 jsou uživatelské pro 2 herní módy, 2 jsou pro 2 umělé inteligence a poslední je pro „střílení“ ve hře Wall Breaker, což je kromě pohybu doleva a doprava jediný akční prvek. Zbývají 3 pictureboxy. Všechny hry a režimy se odehrávají v prvním z nich. 2. je čistě dekorační, protože obsahuje jenom tetris logo a ve 3. se zobrazuje následující TetroBlock, který je uložen v proměnné Shape nextPiece.

Tvorba a design jsou uskutečněny pomocí statické třídy Visual, která odpovídá za správné namalování hrací desky a figurek. Nutnost přemalovat hrací plochy zařídí Paint Event neboli událost, která je vyvolána pomocí funkce PictureBox[13].Invalidate(). Tato funkce na oba živé pictureboxy je volána při každém zmáčknutí každého tlačítka (kromě EXIT tlačítka) a v ve všech událostech timer[12345]\_Tick. Příslušný Paint Event nejprve ověří, že hrací deska i aktivní figurka již byly inicializovány a následně zavolá metodu Visual.DrawGame(ref GameBoard gb, Shape shp, Graphics grafika, Pen tuzka). Tato funkce se skládá z dalších 2 funkcí pro zobrazení hrací desky a aktivní figurky. Pro obě funkce je stěžejní bratrská funkce void DrawRect(Graphics grafika, Pen tuzka, char color, int height, int width). Tato funkce namaluje krásný jeden čtvereček potřebné barvy a umístí ho podle maticových souřadnic height & width do správné polohy v herním poli.

Kapitola 3 – třída shape a jejich 7 dětí

Kapitola 4 – form1.cs aneb třída, co vládne všem a všem káže

Kapitola 5 – nažhavme klávesnice a jdeme na ten tetris

## Část II – Umělá inteligence

kapitola 1 – všechny cesty vedou na hrací desku

kapitola 2 – dobrý tah, nebo zlý, to je, oč tu běží

kapitola 3 – když jde tetroblock na výlet

## Část III – Bonus

kapitola 1 – drtič zdí

# Uživatelská sekce