Zápočtový program PRG2 – Flappy Bird

Zadání

Flappy Bird je hra pro jednoho hráče. Je inspirovaná hrou Flappy Bird, která byla vydána v roce 2013. Tematicky jsem ji trochu upravila, aby symbolizovala studium Bioinformatiky na MFF. Hráč ovládá studenta ptáčka a má za úkol vyhnout se všech překážkách (předmětům BIOINF). V pozadí jsou budovy MFF a cílem je nasbírat co nejvíce bodů, zde kreditů. Po získání 40 kreditů se stává bakalářem.

Uživatelská část

Ovládání

Ptáčka hráč ovládá pomocí mezerníku. Po stisknutí mezerníku ptáček vyskočí. Originálně byla hra na mobilní telefony, kdy ptáček vyskočil po dotyku obrazovky, to je ale pro použití na PC nepraktické.

Úrovně

Hra má 3 úrovně:

1. První ročník – 0-9;
2. Druhák – 10-19;
3. Třeťák – 20-40.

Podle úrovně se určuje rychlost přibližování překážek. Čím vyšší ročník, tím rychlejší překážky jsou.

Cíl

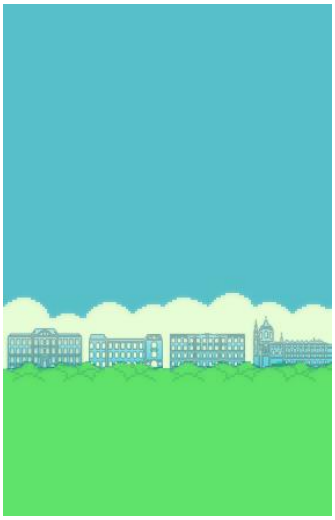
Cílem hry je nasbírat 40 kreditů, tedy vyhnout se 40 překážkám. Po získání 40 kreditů je hra ukončena a ptáček student se stal bakalářem.

Programátorská část

Hra je naprogramovaná pomocí Windows Forms .NET.

Controls

Ptáček, země, uvítací obrazovka, GAME OVER nápis a překážky jsou reprezentován *PictureBoxem*. Ilustrace jsem získala z GitHubu z repositáře Samuela Custodia [1]. Zobrazeny jsou níže. Pro své potřeby jsem některé z nich upravila.



Na gameover obrazovce se zobrazuje *Label* výsledek, který sděluje hráči kam se dostal. Dále *Label* zmackniMe a hratZnovu, které se dotazují, zda chce hráč hrát znovu a jak to má udělat, pokud ano. Skóre, *Label*, se zobrazuje během hry a gameoveru. Sděluje hráči, kolik má momentálně kreditů. *Timer* se stará o běh hry.

Proměnné

Do proměnné *int skoreHrace* se ukládá během stavu HRA (viz níže) počet kreditů, tedy počet překážek, kterým se hráč vyhnul. Během stavu START je vynulována.

Proměnná *int gravitace* určuje o kolik pixelů se ptáček posune na ose x. Bez stisknutí mezerníku je gravitace kladné číslo, tedy ptáček padá. Při stisknutí mezerníku gravitace je stanovena na záporné číslo, tedy ptáček vyskočí, resp. vyletí nahoru. Po uvolnění mezerníku je gravitace nastavena zpět na kladné číslo, aby ptáček opět padal dolů. Gravitace se vždy přičítá odčítá od horní hranice *PictureBoxu* ptáčka.

Proměnná *int posunprekazek* zajišťuje pohyb překážek. Levá hrana PictureBoxu je vždy posunuta o posunprekazek doleva (-= posunprekazek). Zvyšuje se s levely, tedy zrychluje pohyb překážek.

Stavy

Běh hry jsem rozdělila do tří fází:

START – spustí se po inicializaci komponent. Nastavuje defaultní pozice pro překážky a ptáčka, defaultní skóre, gravitaci a pohyb překážek. Zobrazuje zemi, skóre a uvítací obrazovku.
HRA – ve stavu HRA se zobrazí ptáček, překážky a mizí uvítací obrazovka.
KONEC – zviditelňuje GAME OVER nápis, vyzvání k další hře (*Label zmackniMe* a *hratZnovu*) a výsledek hráče.

Events

KeyDownEvent se stará o skok ptáčka, posune ptáčka po ose y nahoru. *Gravitaci* změní na záporné číslo. Pokud dojde ke stisknutí mezerníku ve stavu KONEC, je nastaven stav START. Pokud ve stavu KONEC je stisknut ESC, aplikace je ukončena.

KeyUpEvent nastaví *gravitaci* zpět na kladné číslo, aby po skoku ptáček začal opět padat.

CasovacHryEvent je vlastně game engine celé hry. Game engine běží pouze pokud stav je HRA. Při stavu HRA k horní hranici PictureBoxu ptáčka je přičtena *gravitace*, zajišťuje pád ptáčka. Spouští se metoda *PosouvaniPrekazky* (viz níže). Pak jsou *zeme* a *skore* přivedeny do popředí. Aby je načítající překážky nezakrývaly. Přičítá se skóre tak, aby když hráč mine překážku, přičte se ke kreditům 1. Skóre je zobrazováno. Podle skóre se určuje *rychlostprekazek* dle 3 úrovní. Dále se kontrolují kolize hranice ptáčka s hranicemi překážek, hranicemi *zeme*, a jestli ptáček nevyletěl mimo obrazovku. Pokud došlo ke kolizi, je metoda *PosouvaniPrekazky* zastavena a je nastaven stav KONEC.

Metoda *PosouvaniPrekazky*

Metoda se volá s parametrem index. Pokud je index = 1, jsou *PictureBoxy* překážek přidány do *Listu prekazky*, aby se dalo překážkami iterovat. Všechny překážky jsou posouvány o *rychlostprekazek* doleva. Cyklem se kontroluje, zda se překážky nedostaly mimo obrazovku, pokud ano, horní hranice horní překážky je nastavena na náhodné číslo od -230 do -100, aby se překážky negenerovaly stále stejně vysoko. Překážka je dále posunuta za pravou hranice formuláře. Horní hranice párové dolní překážka se určí dle hodnoty horní hranice horní překážky a je posunuta za formulářem.

Závěr

Při programování Flappy Bird bylo nejtěžší synchronizovat interval Timeru, rychlost padání ptáčka a rychlost překážek. Dále jsem docela dlouho přemýšlela, jak nejsnadněji a nejefektivněji generovat/posouvat překážky. Vyhodnotila jsem, že bude nejlepší mít tři překážky (záleží samozřejmě na šířku formuláře). Nemusím tak řešit generování PictureBoxů a pouze posouvám překážky, když se dostanou mimo okno.

Je to spíše oddychová hra. Hru lze jedině ztížit rychlostí pádu ptáčka nebo posunu překážek, a to je vše. Daly by se přidat další a jiné typy překážet, ale jinak se není kam dál posouvat.

Jako chybu považuji renderování ptáčka, který za sebou nechává bílou stopu. Snažila jsem se to vyřešit bohužel jsem nepřišla na nějaké jednoduché řešení.

Odkazy

[1] https://github.com/samuelcust/flappy-bird-assets/tree/master/sprites