Pacman

Milan Wikarski

Štruktúra súborov

Priečinok `/commands` obsahuje shell scripty pre build and spustenie hry.

Priečinok `/src` obsahuje všetky súbory s C# kódom, obrázky a high score data.

```
src
├── avatars
│   ├── avatar.cs
│   ├── monster.cs
│   └── pacman.cs
├── collectible.cs
├── directed-point.cs
├── events
│   ├── event-revive-monster.cs
│   ├── event-stop-frenzy-mode.cs
│   ├── event-stop-invincibility.cs
│   ├── game-clock.cs
│   └── game-clock-event.cs
├── fps-tracker.cs
├── game.cs
├── highscore.cs
├── keyboard-handler.cs
├── lives-tracker.cs
├── main.cs
├── map.cs
├── score-board.cs
├── settings.cs
├── speed-tracker.cs
└── utils
    ├── array-utils.cs
    ├── box.cs
    ├── direction.cs
    ├── file-manager.cs
    ├── image-set.cs
    ├── linked-list.cs
    └── stopwatch.cs
```

- `avatars/avatar.cs` obsahuje abstraktnú class `Avatar` pre reprezentáciu postavy (tj. pacman alebo príšera)
- `avatars/monster.cs` obsahuje class `Monster : Avatar`, ktorá slúži na reprezentáciu príšerky
- `avatars/pacman.cs` obsahuje class `Pacman : Avatar`, ktorá slúži na reprezentáciu pacmana
- `collectible.cs` obsahuje class `Collectible`, ktorá slúži na reprezentáciu zberateľného bodu alebo bonosuvého predmetu (srdce, čerešňa)
- `directed-point.cs` obsahuje class `DirectedPoint`. To je 2D bod (má vlastnosti left, top), ktoré je rozšírený o direction. Tá je rozhodujúca v prípade kolízií - kolízie sa zisťujú rozlične pre jednotlivé smery.
- `events/game-clock.cs` obsahuje class `GameClock`. Tá slúži na plánovanie a spúšťanie eventov. Ďalej sleduje čas, ktorý ubehol od posledného tick-u a priemernú dĺžku trvania tick-u.
- `events/*.cs` ostatné súbory v priečinku `events` sú class-y, ktoré slúžia na reprezentáciu jednotlivých eventov. Každý event musí dediť od `abstract class GameClockEvent` a musí implementovať `void Execute()`.

- `fps-tracker.cs` obsahuje class `FpsTracker : Label`. Slúži na sledovanie priemerneho hodnoty FPS (frames per second; tj. počet volaní metódy `void Tick` za sekundu) za posledných 1000ms.
- `game.cs` obsahuje class `Game`, ktorá združuje všetky prvky hry.
- `highscore.cs` obsahuje class `Highscore`, ktorá umožňuje prácu s highscore (čítanie, písanie).
- `keyboard-handler.cs` obsahuje class `KeyboardHandler : TextBox`. Slúži na sledovanie klávesnice.
- `lives-tracker.cs` obsahuje class `LivesTracker : Label`, ktorá spravuje a zobrazuje životy pacmana.
- `main.cs` obsahuje implementáciu metódy `void Main()`. Jediné, čo sa v tomto súbore deje je vytvorenie instance hry (`Game`) a jej spustenie (`Game.Run()`).
- `map.cs` obsahuje class `Map : PictureBox`. Slúži na zobrazenie mapy a združuje všetky významné body.
- `score-board.cs` obsahuje class `ScoreBoard : Label`, ktorá slúži na sledovanie a zobrazovanie skóre.
- `settings.cs` obsahuje nastavenia hry (rýchlosť hráča, počet príšer, dĺžku nesmrteľnosti po kolízii)
- `speed-tracker.cs` obsahuje class `SpeedTracker : Label`, ktorá sleduje priemernú rýchlosť hráča v px/s za posledných 1000ms.
- `utils/*` obsahuje rôzne utility classy a funkcie, ktoré sa používajú naprieč programom

Obrazovky

Hrá ma 3 obrazovky, ktoré sú implementované v `class Game`. Každá obrazovka má pridelený stav (tzn. podľa stavu sa dá rozoznať, ktorá obrazovka je aktívna) a má vlastnú metódu, pomocou ktorej sa vykreslí obsah, nabindujú sa eventy a poprípade sa vykanajú ďalšie akcie.

- **WelcomeScreen** je prvá obrazovka, ktorú užívateľ pri spustení hry uvidí. Obsahuje banner a tlačidlo pre spustenie novej hry.
- **GameScreen** je herná obrazovka. Ak je aktívna, metóda `Tick` sa spúšťa vo (viacmenej) pravidelných intervaloch.
- **GameOverScreen** je obrazovka, ktorá sa spustí po skončení hry, tj. keď hráč príde o posledný život. Obsahuje správu "GAME OVER", informáciu o počte získaných bodov, high score a tlačidlo, ktorý sa hráč presunie opäť na *WelcomeScreen*.

Medzi obrazovkami sa dá prechádzať iba cyklicky, pomocou metódy `NextState`. Tzn. jediný spôsob, ako medzi nimi prechádzať je tento:

```
WelcomeScreen --> GameScreen --> GameOverScreen --> WelcomeScreen --> ...
```

Každá obrazovka je zodpovedná za to, aby do spojového zoznamu `toDispose` pridala tie WinForms elementy, ktoré by mali po prechode na ďalšiu obrazovku zaniknúť.

Spustenie hry

Obrazovky **WelcomScreen** a **GameOverScreen** sú statické, tzn. čakajú na input od usera, ktorý spustí metódu `NextState`.

Ak je aktívna obrazovka **GameScreen**, hra je spustená. Všetko sa vykonáva v metóde `Tick`. Pred prvým spustením metódy `Tick` sa ale vykoná inicializácia. Tá je implementovaná priamo v metóde `GameScreen` a obsahuje:

- inicializáciu displejov (ScoreBoard, LivesTracker, FPSTracker, SpeedTracker)
- Inicializáciu mapy
- Inicializáciu pacmana a príšeriek
- Prvotné vykreslenie všetkých objektov
- Spustenie hodiniek

Main Loop

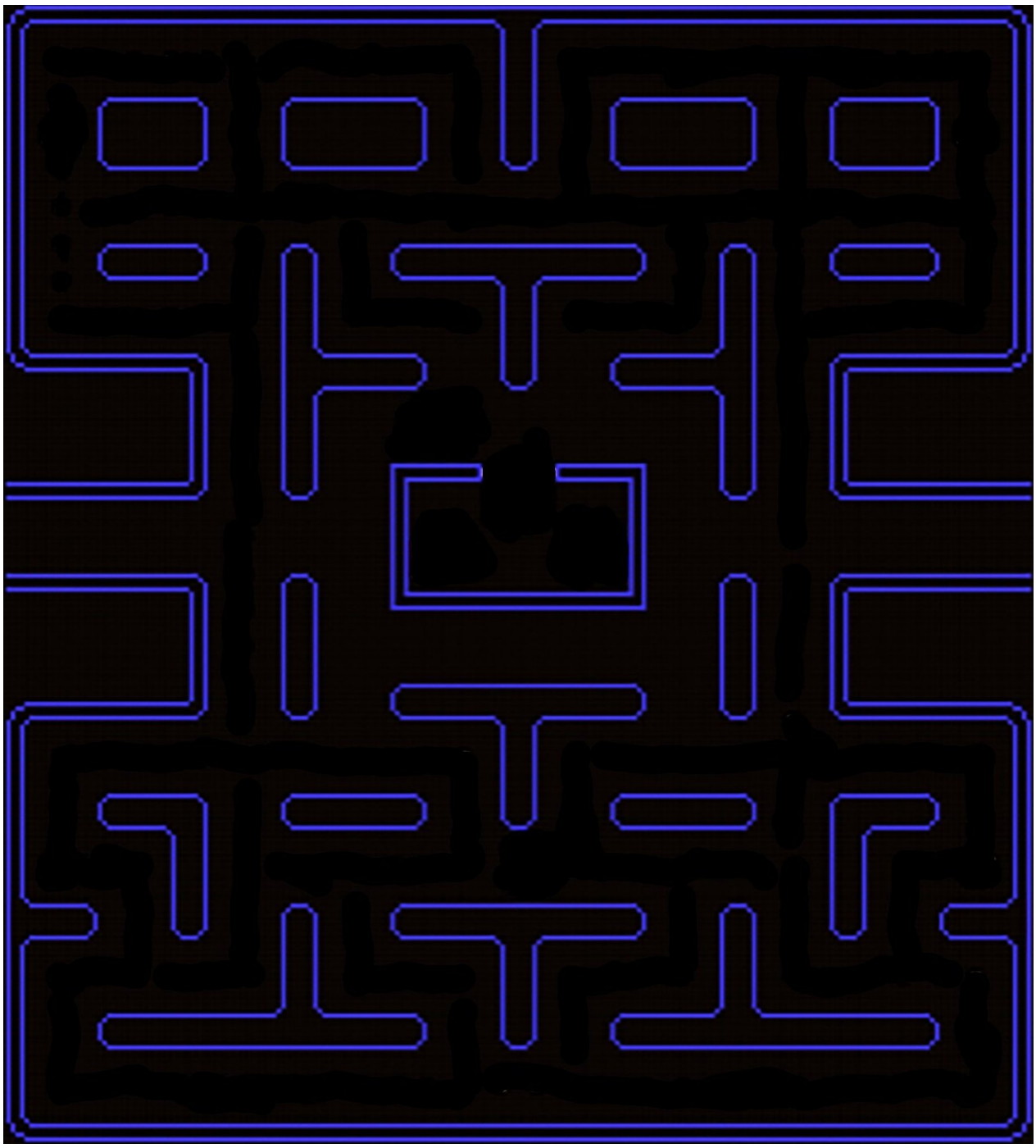
Po inicializácii hry sa vo (viacmenej) pravidelných intervaloch spúšťa metóda `Tick`. Tá má plnú zodpovednosť za obsluhu hry (Ak iné metódy zasahujú do stavu hry, potom musia byť volané v metóde `Tick` alebo v metóde, ktorá je volána z metódy `Tick`, …).

V každej iterácii hlavného cyklu sa:

- zaznamená spustenie metódy `Tick` (na to slúži `GameClock`),
- vykonajú všetky eventy, ktoré sa už môžu vykonať,
- vykoná sa pohyb hráča (tj. posun vpred a prípadná rotácia, ak je možná),
- vykoná sa pohyb príšer (viz. poznámka hore),
- kontrola kolízií so zberateľnými bodmi a bonusmi
- kontrola kolízií s príšerami,
- update FPS
- update sledovanej rýchlosti

Mapa

Mapa, ktorá sa v hre používa je na nasledujúcom obrázku:



mapa sa skladá z 10 riadkov a 10 stĺpcov. Jednotlivé stĺpce a riadku sú prerušené prekážkami, z toho vznikajú chodby, po ktorých sa môžu postavy presúvať.

Prekážky

Implementácia chodieb a prekážok je spravená pomocou `DirectedPoint`. Tj. mapa obsahuje význačné body, na ktorá postava reaguje v prípade kolízie. Existujú 2 typy význačných bodov:

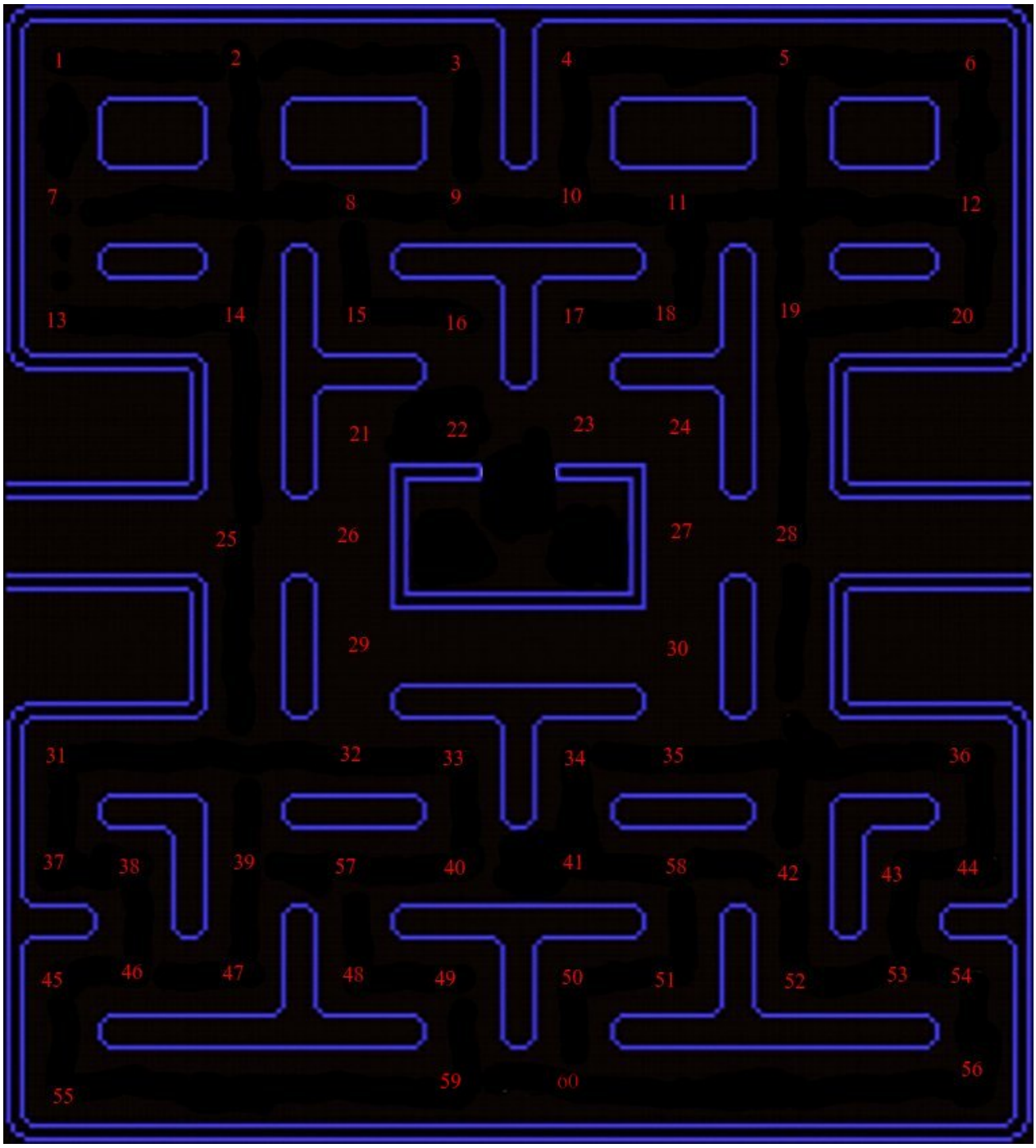
- kolízne body (stopping points)
- rozačné body (turning points)

Kolízne body

Postavy sa môžu voľne pohybovať po mape. V hre nie sú implementované prekážky ako 2D objekty. Namiesto toho mapa obsahuje kolízne body. Ak postava príde na kolízny bod (tj. postava má kolíziu s kolíznym bodom) zo správneho smeru, zastaví sa. Takto sú vlastne implementované steny, na ktoré postava narazí, a tým sa jej pohyb dočasne zastaví.

Kolízny bod môže mať rôzne smery. Kolízny bod, ktorý je v rohu (napr. 20) obsahuje dva smery (v prípade 20 dole a napravo). Kolízny bod, ktorý nie je v rohu (napr. 59) obsahuje jeden smer (v prípade 59 dole).

Mapa obsahuje 60 kolíznych bodov:

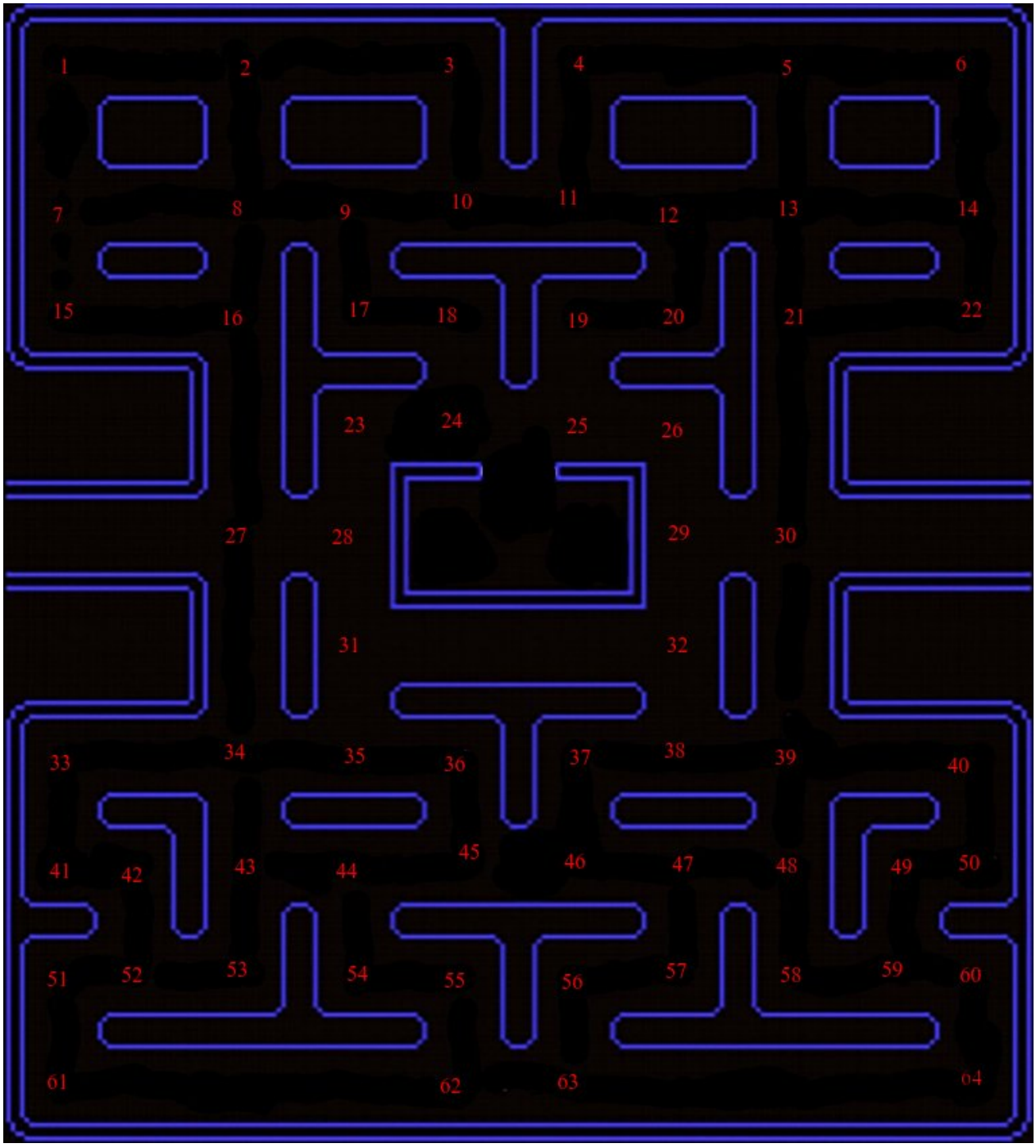


Rotačné body

Každá postava má smer pohybu (hore, dole, napravo, naľavo). Pohyb potom znamená prirátanie alebo odrátanie istej hodnoty k/od `Left` alebo `Top`. Vždy sa môže otoriť do opačného smeru (O 180 stupňov). Ostatné rotácie sú povolené iba na špeciálnych miestach. Tieto miesta sa nazývajú rotačné body.

Rotačné body sú implementované pomocou class `DirectedPoint`. Majú teda špecifikovaný smer (jeden alebo viac). Kolízie s nimi teda závisia od smeru, ktorým postava putuje. Otočiť sa potom postava môže iba do smeru opačného k smerom, ktoré rotačný bod obsahuje (napr. bod 33 obsahuje smery naľavo a hore, a teda postava sa tam môže otočiť iba doprava a dole. Bod 13 obsahuje všetky smery, takže sa postava môže otočiť do ľubovoľného smeru).

Mapa obsahuje 64 rotačných bodov:



Postavy

Jediné pohyblivé objekty v hre sú pacman a príšerky. Oba objekty sú instancie class `Avatar`. Tá ma implementované metódy pre pohyb a rotáciu a udržiava aktuálnu pozíciu postavy.

Pozícia

Pozícia je udržiavaná pomocou class `Box`, ktorá obsahuje údaje `Left`, `Top`, `Width` a `Height`. Všetky tieto údaje sú typu `double` (narozdiel od Windows Forms, kde sú tieto údaje typu `int`). Pri vykresľovaní sa údaje najprv zaokrúhlia, a potom konvertujú na `int`. Kolízie sa ale počítajú s údajmy typu `double`. (Tento prístup je nutný, pretože rýchlosť postavy musí byť nezávislá od počtu FPS, viac o tom v časti *Pohyb*).

Pohyb

Pohyb je vždy iba v smere, ktorým je avatar otočený. Pre pohyb sú implementované 2 metódy:

- `CanMove`, ktorá slúži na zistenie, či sa postava ešte môže pohnúť dopredu. Prejde cez všetky kolízne body a zistí, či niekde postava "nenarazí do steny", ak by sa pohla ďalej
- `Move`, ktorá slúži na posunutie postavy o istú jednotku vzdialenosti.

Rýchlosť pohybu postavy je nastavená v `settings.cs`, kde je špecifikovaná v pixeloch za milisekundu. DĺŽKa posunu v danej iterácii hlavného cyklu sa potom vypočíta podľa doby, ktorá prešla od minulej iterácie (na to slúži `GameClock`).

Metódy `CanMove` a `Move` preto majú argument `double multiplier`. To je počet milisekúnd od predošlej iterácie. Týmto číslom sa prenásobý základná rýchlosť postavy. Okrem toho existuje aj horný limit pre hodnotu `multiplier`, aby nenastali príliš veľké skoky, ktoré by mohli spôsobyť "vypadnutie" postavy z mapy.

Rotácie

Rotácie boli opísané v časti *Mapa >> Rotačné body*. Rotácie pacmana sú pomocou klávesnice. Príšery vždy zmenia smer, ak narazia na kolízny bod. Ak sa nachádzajú na rotačnom bode ale nie na kolíznom bode, potom zmenia smer s pravdepodobnosťou 50%.

Kolízia pacmana s príšerou

V každej iterácii hlavného cyklu sa kontroluje kolízia pacmana s príšerou. Ak takáto kolízia nastane, príšera zomrie (zmizne z mapy a o krátky čas sa opäť objaví na svojej pôvodnej pozícii). Hráč potom stratí život a pacman sa stane dočasne nesmrteľným. Kolízie pacmana s príšerou sa počas jeho nesmrteľnosti nekontrolujú.

Ovládanie

Pacman sa vždy pohybuje v smere, ktorým je otočený, dokým nenarazí do steny. Hráč môže ovládať iba jeho rotácie. Ovládanie je pomocou šípiek. Komunikácia s klávesnicou (čítanie) je implementované v class `KeyboardHandler`.

Namiesto toho, aby hra vždy reagovala na stlačenie klávesy, si `KeyboardHandler` zapamätá poslednú stlačenú klávesu a čas, kedy bola stlačená. Potom po istú dobu (400ms, dá sa zmeniť) v každej iterácii hlavného cyklu sa skúša vykonať rotácia do daného smeru. Takto môže hráč vo veľmi krátom predstihu rotovať pacmana a tak zabezpečiť plynulý pohyb.

Body a bonusy

Na začiatku sa v pravidelných rozostupoch (17px) zobrazia na mape body, ktoré musí hráč zbierať. Každý bod má istú šancu, že sa z neho stane bonus. Sú teda tri typy bodov:

- obyčajný bod
- čerešnička
- život

Obyčajný bod

Ak pacman zoberie obyčajný bod, obrázok bodu zmizne a hráčovi sa zvýši skóre o 1.

Čerešnička

Keď pacman zoberie čerešničku, vstúpy do *frenzy mode*. Vtedy kolízia pacamana s príšerou spôsobý, že ju pacman "zožerie". Tak príšere dočasne zmizne a hráčovi sa priráza 50 bodov. Pacman je počas frenty módu 1.2x rýchlejší.

Život

Keď pacman zoberie život, priráta sa mu jeden život.

Koniec hry

Hra končí, keď hráč stratí posledný život. Potom sa hra presunie do ďalšieho stavu a spustí sa metóda `GameOverScreen`

Známe nedostatky

V tejto časti sa nachádza zoznam známych nedostatkov

Nedokonalé zarovnanie postáv

Pozícia postavy v dvojrozmernej sústave je reprezentovaná dvoma číslami: `double Left` a `double Top`, ktoré sú súčasťou class `Box`. Mapa sa skladá z 10 riadkov a 10 stĺpcov. V každom okamihu sa môže postava pohybovať iba po jednom riadku (vtedy sa pohybuje horiznotálne) alebo po jednom stĺpci (vtedy sa pohybuje vertikálne). V rotačných bodoch sa môže zmeniť stĺpec a riadok, na po ktorom sa postava pohybuje.

V takom prípade ale môže nastať, že sa postava nedostane presne do stredu tohoto stĺpca alebo riadku. Takto vznikne odchylka veľkosti pár pixelov. Po stránke hrateľnosti to nepredstavuje žiaden problém, ale vizuálny zážitok by sa zlepšil odstránením tohoto nedostatku.

Nedokonalé zarovnanie bodov

Veľkosť mapy bola vybraná tak, aby sa zmestila do okna FULL HD obrazovky. Body, ktoré sa na mape nachádzajú sú v pravidelných intervaloch veľkosti 17px. Na miestach, kde sa krížujú riadky a stĺpce sa niekedy neprekryjú body. Potom vzniknú miesta, kde body nie sú v rozostupoch presne 17px. Čiastočne som tento problém vyriešil vynechaním niektorých bodov z mapy. Vizuálne vyzerá všetko vporiadku. Toto je teda problém, ktorý by trápil skôr pedantného užívateľa.

Nižšie FPS pri vykresľovaní

Windows Forms nie je najlepší spôsob, akým riešiť vykresľovanie postáv. Hra má iba 5 pohiblivých postáv, no ak sa hýbu všetky naraz (tzn. 5 instancií `PictureBox` mení svoju pozíciu, a teda sa prekresľujú), FPS klesna na ~40. Vizuálne to nie je nepríjemne, ale nie je to optimálnych 60.

Experimentálne som zistil, že za zníženie FPS môže **iba** vykresľovanie. Ak sa vykresľovanie zruší (tzn. postavy sa hýbu, kontrolujú sa kolízie, atď ale instancie `PictureBox` sú stále na tom istop mieste, a teda sa nevykresľujú nanovo), FPS ostáva stabilne na 60.

Možná optimalizácia

V tejto časti sa nachádza zoznam možných vylepšení. Vylepšenia, ktoré by odstránili známe nedostatky tu neuvádzam, aby som sa zbytočne neopakoval.

Kontrola kolízií

V každej iterácii sa kontrolujú kolízie:

- Pacman s 60 kolíznymi bodmi (60)
- Pacman s 64 rotačnými bodmi (64)
- Pacman s 503 zberateľnými bodmi (503)
- Pacmas s 4 príšerkami (4)
- 4 príšerky s 60 kolíznymi bodmi (240)
- 4 príšerky s 64 kolíznymi bodmi (256)

Čo je dokopy ~1100 kontrol (niektoré sa preskočia, ale stále je to aspoň jedno `if`). Ako som spomínal v časti *Známe nedostatky >> Nedokonalé zarovnanie postáv*, každý objekt, ktorý sa na mape nachádza sa môže nachádzať iba v jednom stĺpci a riadku. Počet kontrol by sa dal znížiť tak, že by sa vždy kontrolovali iba kolízie s bodmi v rovnakom riadku a rovnakom stĺpci, ako postava. Tak by sa počet kontrol znížil niekoľkonásobne.