Custom A\* Pathfinding v herním engine GODOT

# Cíl projektu

GODOT Engine má vlastní integrované varianty algoritmu A\*, ty ale fungují buď na spojitém prostoru nebo pouze na čtvercové mřížce.

Cílem tohoto projektu je vytvořit vlastní variantu algoritmu A\* která by mohla být použita i např. na izometrickou mřížku a po případné úpravě prohledávání sousedů i na prohledávání hexagonální mřížky.

Obsah obrázku vzor, design, látka, snímek obrazovky

Popis byl vytvořen automaticky

Obrázek 1 Čtvercová mřížka

Obsah obrázku vzor, snímek obrazovky, design, Barevnost

Popis byl vytvořen automaticky

Obrázek 2 Izometrická mřížka

Obsah obrázku vzor, Symetrie, design, snímek obrazovky

Popis byl vytvořen automaticky

Obrázek 3 Hexagonální mřížka

# GODOT Engine



Godot Engine je bezplatný a open source software vydaný pod licencí MIT.

Jako programovací jazyk využívá primárně vlastní jazyk GDScript. Ten je syntaxem velice podobný jazyku Python, na rozdíl od něj ale vyžaduje explicitní deklaraci proměnných.

Další možností je celá samostatná verze Godot, která pracuje s jazykem C#.

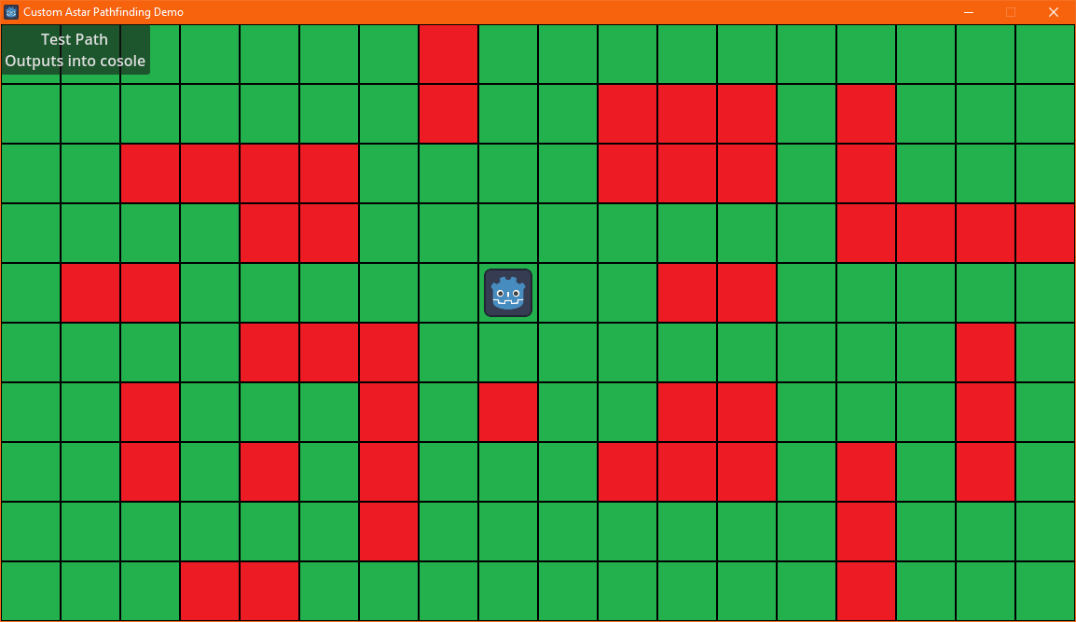
# A\* Algoritmus

* Pro prohledávání stavového prostoru se používá seznamů OPEN a CLOSED
  + Seznam OPEN je seznam uzlů k prohledávání
  + Seznam CLOSED je seznam již prohledaných uzlů
* Pro určení toho, který uzel prohledávat se využívá hodnotící funkce:
  + *f* – celková hodnota ceny uzlu
  + *g* – hodnota ceny přechodu z počátečního uzlu do aktuálního
    - *c(i, j)* – cena přechodu z uzlu *i* do uzlu *j*
  + *h* – heuristická funkce určující většinou vzdálenost od cílového uzlu

## Popis algoritmu:

1. Zapsání počátečního stavu do seznamu OPEN
2. Pokud je seznam OPEN prázdný neexistuje řešení a prohledávání je ukončeno
3. Vybrání stavu *i* s nejmenší hodnotou *f(i)* ze seznamu OPEN. V případě většího množství stavů se stejnou hodnotou *c* zkontrolovat, zda není některý z nich cílový stav, pokud ano vybrat jej
4. Přesun vybraného uzlu z OPEN do CLOSED (vymazání z OPEN)
5. Je-li stav *i* cílový ukončit prohledávání – řešení nalezeno
6. Expanze stavu *i*. Pro každého následníka stavu i vypočítat hodnotu f
   * Pokud stav není ani v OPEN ani v CLOSED zařadit jej do OPEN
   * Pokud je stav j v OPEN nebo CLOSED ale s hodnotou f vyšší než právě vypočtená přepsat tuto hodnotu na právě vypočtenou a změnit jeho rodiče na aktuální uzel i. Přitom pokud je stav j v CLOSED přesunout jej zpět do OPEN a vymazat z CLOSED
7. Pokračování krokem č. 2.

# Uživatelský popis aplikace



Obrázek 4 Ukázka spuštěné "hry"



Obrázek 5 "postavička" agenta

### Popis obsahu okna

* Agent – „postavička“ která se pohybuje po cestě nalezené A\*.
* Tlačítko „Test Path“ – spouští prohledávání mezi pevně nastavenými body (0, 0) a (15, 7), indexování začíná od 0 z levého horního rohu. Výstup se nepromítne na pohybu agenta, ale pouze se vypíše do konzole. Pro zobrazení konzole spusťte *Custom\_Astar\_Demo.console.exe.*
* Zelená pole – Pole volná pro pohyb.
* Červená pole – Překážky.

### Ovládání dema

Ovládání je realizováno pouze levým tlačítkem myši, při kliknutí na jakékoliv pole se spustí prohledávání a pokud je nalezena cesta spustí se pohyb agenta.

# Obsah obrázku text, snímek obrazovky, číslo, Písmo Popis byl vytvořen automatickyPopis kódu

Program obsahuje 4 skripty psané v jazyku GDScript:

* gui.gd – připojen k GUI.tscn
* Custom\_AStar.gd - nepřipojen
* environment.gd – připojen k Environment.tscn
* unit.gd – připojen k Unit.tscn

Obrázek 6 Strom prvků scény programu

Tyto skripty jsou připojovány k předefinovaným objektům, a dědí tím jeho vlastnosti a funkce.

### gui.gd

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

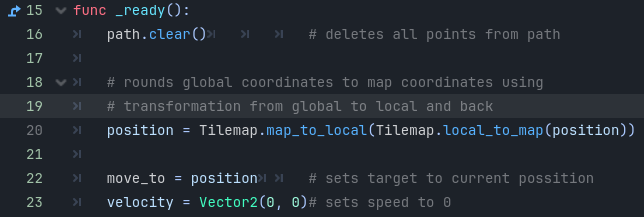
* extends je funkce pro předání parametrů a funkcí objektu CanvasLayer do skriptu
* @onready je funkce která se spouští pokaždé když je daný objekt načten do aktuální scény v tomto případě se do proměnné env přenáší reference na celý objekt Environment.tscn ($“..“ je ukazatel kořenový prvek ve stromu prvků)
* Funkce \_on\_button\_pressed() je spuštěna po přijetí signálu z tlačítka „Test Path“ a spouští testovací funkci objektu Environment.tscn.

### unit.gd

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automaticky

* Extends – předání dat z objektu CharacterBody2D
* @onready var Tilemap – Tile map je objekt pro vytváření mřížky polí ať už se jedná o terén nebo jen pozadí. Zde je reference na tento objekt přiřazena proměnné pro pozdější použití
* Const – konstanta (nelze upravovat v kódu, pouze číst)
* Var – promněnná (lze upravovat uvnitř kódu)
* SPEED – rychlost pohybu agenta
* Min\_target – minimální vzdálenost cíle pro spuštění hledání cesty
* Eps – tolerance odchylky od cílového bodu, aby byl uvažován jako dosažený
* Path – list vektorů pozic cesty do cílového bodu
* Move\_to / to – vektory souřadnic pro aktuální bod cesty



* Path.clear() – funkce clear() je interní funkcí pro Array a odstraní všechny prvky tohoto listu
* Position – pozice objektu unit v pixelech v prostoru scény
* Map\_to\_local / local\_to\_map transformují souřadnice z lokálních souřadnic scény v pixelech na souřadnice v objektu Tilemap (každé pole má souřadnice jako by bylo jeden pixel) a zpět
* Move\_to = position nastavuje jako cíl pohybu aktuální pozici (zabránění náhodného pohybu při startu)
* Velocity = … - nastavení rychlosti pohybu na 0

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Ř. 30 – pokud je vzdálenost od cíle větší než nastavené minimum nastavit hodnotu rychlosti na daný směr k cíli a danou velikost (rychlost je 2 složkový vektor x a y)
* Ř. 33 – interní fyzikální funkce pro pohyb těles
* Ř. 36 – kontrola dosažení cíle
* Ř. 37 – pokud je list cesty prázdná spustí funkci \_ready() která resetuje a zastaví pohyb agenta
* Ř. 39 – pokud cesta není prázdná nastaví další její bod jako nový cíl

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Funkce move() vybírá první prvek ze seznamu cesty a přiřadí ho proměnné to kterou vrací jako výstup. Pokud je cesta prázdná nastaví cíl na aktuální pozici agenta
* Funkce debug() vypisuje do konzole cestu přijatou z algoritmu A\*

### environment.gd



* extends – předání parametrů a funkcí objektu
* tilemap – objekt mapy mřížky polí scény
* unit – předání objektu Unit.tscn ze scény do proměnné
* Astar = preload(…).new() – preload načte objekt v závorkách do paměti, v tomto případě jde o skript algoritmu A\*, a .new() vytvoří jeho instanci. Ta je poté předána do proměnné Astar
* path – list pro body cesty
* DEBUG TOOLS – proměnné pro ovládání debug režimu skriptu

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automaticky

* \_ready() – spuštěna při načtení do paměti
* Ř. 18 – nastavení pohybového režimu agenta tak aby korespondoval s pohybem při pohledu ze shora

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Test\_function() – funkce spouštěná z objektu GUI tlačítkem nikoliv kliknutím do prostoru. Jsou v ní pevně nastavené polohy startu a cíle cesty a funguje pro otestování funkčnosti algoritmu A\* aniž by byl potřeba agent. Její výstup je vypsán pouze do konzole

Obsah obrázku text, snímek obrazovky, software, Multimediální software

Popis byl vytvořen automaticky

* \_unhadeled\_input(event) - hlavní funkce tohoto skriptu. Je spouštěna automaticky interně při jakékoliv interakci s oknem programu.
* Ř. 34 a 35 – kontrola jaký vstup byl aktivován, v tomto případě, zda bylo stisknuto levé tlačítko myši
* Ř. 36 – získání lokální pozice kurzoru myši a její zapsání do proměnné target
* Ř. 37 a 38 – vymazání veškerého obsahu listu cesty a následné spuštění algoritmu A\*. startovací bod je pozice agenta a cílový bod jsou souřadnice kurzoru
* Ř. 40–43 – pokud je aktivní debug režim výstup A\* jsou dva listy. Zde jsou tyto listy odděleny
* Ř. 45 a 46 – předání listu cesty agentu a spuštění jeho funkce debug()

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automaticky

* Při debug režimu je výstup A\* proměnná dvou listů cesty. Na první pozici je cesta v lokálních souřadnicích, která je poté předána agentu. Na druhé je dvousložkový list všech uzlů cesty, první složka je souřadnice uzlu a druhá souřadnice rodiče tohoto uzlu

### Custom\_AStar.gd

Obsah obrázku text, snímek obrazovky

Popis byl vytvořen automaticky

* Extends – předání parametrů a funkcí obecného objektu „Object“ do tohoto skriptu
* Class\_name – nastavení jména objektu reprezentovaného tímto skriptem
* Open – list uzlů k prohledání
* Closed – list již prohledaných uzlů
* Path – list uzlů cesty i s jejich rodiči
* Tile\_cost – cena přechodu na tento uzel (pro toto demo jsou všechny oceněny na 1, v praxi by např. bažina mohla mít ohodnocení 5 a beton např. 2)
* F – celková hodnota uzlu
* G – dosavadní hodnota pro cestu do tohoto uzlu
* H – heuristická hodnota ceny, zde je použita Euklidovská vzdálenost od cíle
* Start\_point – souřadnice startovního bodu prohledávání
* End\_point – souřadnice cílového bodu prohledávání
* Coords – univerzální vektor pro přenášení souřadnic bodů
* Walkable – proměnná pro rozlišení dosažitelných uzlů od překážek
* Parent\_coords – univerzální proměnná pro souřadnice rodiče uzlu
* Tile – „objekt“ pro skladování dat jednotlivých uzlů. Hodnoty indexů od 1. jsou následovné:
  1. f – celková cena uzlu – použito pro výběr uzlu k expanzi
  2. g – dosavadní cena cesty do tohoto uzlu
  3. tile\_cost – cena přechodu na tento uzel
  4. coords – souřadnice uzlu
  5. parent\_coords – souřadnice rodiče tohoto uzlu
  6. walkable – dosažitelnost uzlu
* Tile\_map – proměnná pro objekt mapy polí mřížky

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Search\_array – list pro prohledávání sousedů, tento funguje pro čtvercovou a izometrickou mřížku
* Tile\_state – hromadná generace proměnných pro dosažitelnost uzlu

Obsah obrázku text, snímek obrazovky, Písmo, software

Popis byl vytvořen automaticky

* funkce clear() resetuje všechny výše popsané proměnné na základní hodnoty

Obsah obrázku text, snímek obrazovky

Popis byl vytvořen automaticky

* tato funkce „vytahuje“ data z objektu Tile\_map, a to konkrétně dosažitelnost daného pole. Dále zapisuje do proměnné Tile základní hodnoty pro dané pole
* atlas\_coords – Atlas je grafický soubor (např. PNG) ve kterém jsou vedle sebe umístěny jednotlivá pole. Souřadnice atlasu mají x a y složku, v našem případě má dosažitelné pole souřadnice (0, 0) a překážka (1, 0)

Obsah obrázku Barevnost, Obdélník, čtverec, zelené

Popis byl vytvořen automaticky

Obrázek 7 Atlas polí pro toto demo (zelené pole – dosažitelné, červené - překážka)

* aby nedocházelo k referování a poté přepisování již existujících hodnot polí tak tato funkce vrací kopii proměnné Tile pomocí integrované funkce duplicate()

Obsah obrázku text, snímek obrazovky, Písmo, software

Popis byl vytvořen automaticky

* Tato funkce prohledává sousedy právě expandovaného uzlu. Pro toto demo je pevně nastaveno 8 sousedů (4 přímí a 4 diagonálně)
* Neighbors\_coords – list souřadnic sousedů
* Neighbors – list vlastních proměnných Tile
* Neighbors\_out – list uzlů které budou vráceny touto funkcí, není nutné vracet uzly které jsou překážky
* Ř. 80 – cyklus pro postupné získání proměnné Tile pro všechny sousedy
* Ř. 84 – cyklus pro vyfiltrování nedosažitelných uzlů

Obsah obrázku text, snímek obrazovky, software

Popis byl vytvořen automaticky

* Tato funkce vypočítá hodnoty ohodnocení f a g pro daný uzel a vrátí tyto hodnoty pomocí pomocného vektoru transfer
* Ř. 93-99 seřazení a příprava souřadnic pro výpočet heuristické funkce.
* H – výpočet heuristické funkce, zde je to Euklidovská vzdálenost
* G – výpočet nové ceny tohoto uzlu
* F – celková cena uzlu

#### Hlavní funkce A\* - Find\_path

Obsah obrázku text, snímek obrazovky, software

Popis byl vytvořen automaticky

* Ř. 116 – resetování všech proměnných
* expanded\_tile – proměnná pro právě expandovaný uzel
* Ř. 120-137 – ze vstupních souřadnic startovního a koncového uzlu se získají a zapíšou jejich vlastnosti, startovní uzel se navíc zapíše do seznamu OPEN
* Ř. 139 – podmínka ukončení hledání v případě že koncový uzel je překážka

#### Začátek hlavního cyklu A\*

Obsah obrázku text, snímek obrazovky, software, Multimediální software

Popis byl vytvořen automaticky

* Hlavní while cyklus je opakován dokud není seznam OPEN prázdný nebo dokud není ukončen vnitřní podmínkou
* První krok je seřazení seznamu OPEN, jelikož je hodnota f jako první pro každý prvek seznamu je tento seznam seřazen podle této hodnoty
* Ř. 144-154 – kontrola seznamu OPEN na větší počet stejně ohodnocených uzlů, pokud je jeden z nich koncový stav bude vybrán ten, pokud ne je vybrán první prvek seznamu OPEN
* Ř. 158-162 – kontrola ukončovací podmínky. Pokud je právě expandovaný uzel koncový uzel prohledávání je ukončeno
* Ř. 165 – expanze aktuálního uzlu funkcí Search\_neighbors

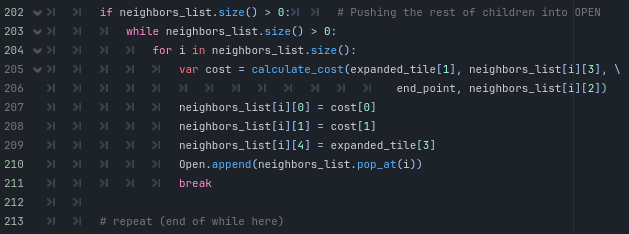
Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Kontrola přítomnosti sousedů expandovaného uzlu v seznamu OPEN
* Pokud je nová vypočtená cena souseda menší než ta která je v seznamu OPEN nastaví se expandovaný uzel jako rodič tohoto souseda a takto upravený se nahradí v seznamu OPEN a odebere se ze seznamu sousedů.
* Pokud nová vypočtená cena není menší než již existující v seznamu OPEN tento soused se odebere ze seznamu sousedů



* Zde se provádí identická kontrola, jako pro seznam OPEN, pro seznam CLOSED



* Zapsání zbylých sousedů, kteří nejsou ani v OPEN ani v CLOSED do seznamu OPEN
* Poté je konec hlavního While cyklu – opakování, dokud není vnitřně ukončen

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automaticky

* Od řádku 215 do konce skriptu je zařazování uzlů do listu cesty a následné vrácení hodnoty této cesty do skriptu ze kterého byla zavolána
* Ř.216-220 – prohledání listu CLOSED pro koncový uzel, při jeho nalezení se zapíše do listu cesty jako první (na konec)

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automaticky

* Prohledání seznamu CLOSED a postupné řazení uzlů do seznamu cesty podle jejich rodičovského uzlu, toto prohledávání probíhá dokud není na začátku seznamu startovací uzel

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automaticky

* Path – seznam 2 složkových prvků [souřadnice uzlu, souřadnice rodiče]
* Path\_out – seznam souřadnic jednotlivých uzlů
* Ř.230-232 – transformace souřadnic ze souřadnic mapy na lokální a zároveň přenášení souřadnic uzlů, už bez souřadnic rodičů, do seznamu Path\_out
* Ř.234-239 – pokud je použito debug funkce je vrácen list listů ve kterém jsou obsaženy jak list s rodiči, tak list lokálních souřadnic bez rodičů. Pokud není použito debug funkce je vrácen pouze list lokálních souřadnic bez rodičů

# Zdroje

A\* - Přednášky od Pana doktora RNDr. Jiří Dvořák, CSc.

Godot engine - <https://godotengine.org/> (celý Engine verze 4.2 má velikost 103 MB a je v podobě bez-instalační aplikace)

GDScript dokumetace - <https://docs.godotengine.org/en/stable/>