Gymnázium, Praha 6, Arabská 14
Obor programování

Ročníkový projekt

Plošinová videohra za pomoci LWJGL

Vojtěch Nejedlý

duben 2024

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 8. dubna 2024 Vojtěch Nejedlý

Anotace

Cílem mé práce bylo vytvoření plošinové 2D hry s pohledem z boku s vykreslovaním na grafické kartě za pomoci Javy. V dokumentaci práce najdete popis a využití jednotlivých tříd, některých metod a editoru hry.

Obsah

1. Úvod

Oproti práci z druhého ročníku, kde jsem sice také pracoval na vytvoření 2D hry, ale moc se mi nelíbil můj editor, jsem se rozhodl, že se pokusím vytvořit jeden více komplexní. Také se mi nechtělo třetí rok tvořit něco v JavaFX a chtěl jsem se ponořit do větší hloubky.

1.1 Popis hry

Cílem hráče je překonávat překážky a bez vypadnutí z mapy se dostat do cíle. Hráč má možnost si vytvořit vlastní mapu a například ji sdílet s ostatními.

1.2 Použité nástroje

Projekt jsem vytvořil IntelliJ IDEA Community Edition od společnosti JetBrains. Vývojové prostředí podporuje Windows, macOS i Linux.

Obrázky (textury) do hry jsem vytvářel v programu Aseprite, který si jde zaplatit na platformě Steam nebo bezplatně postavit z github repositáře.

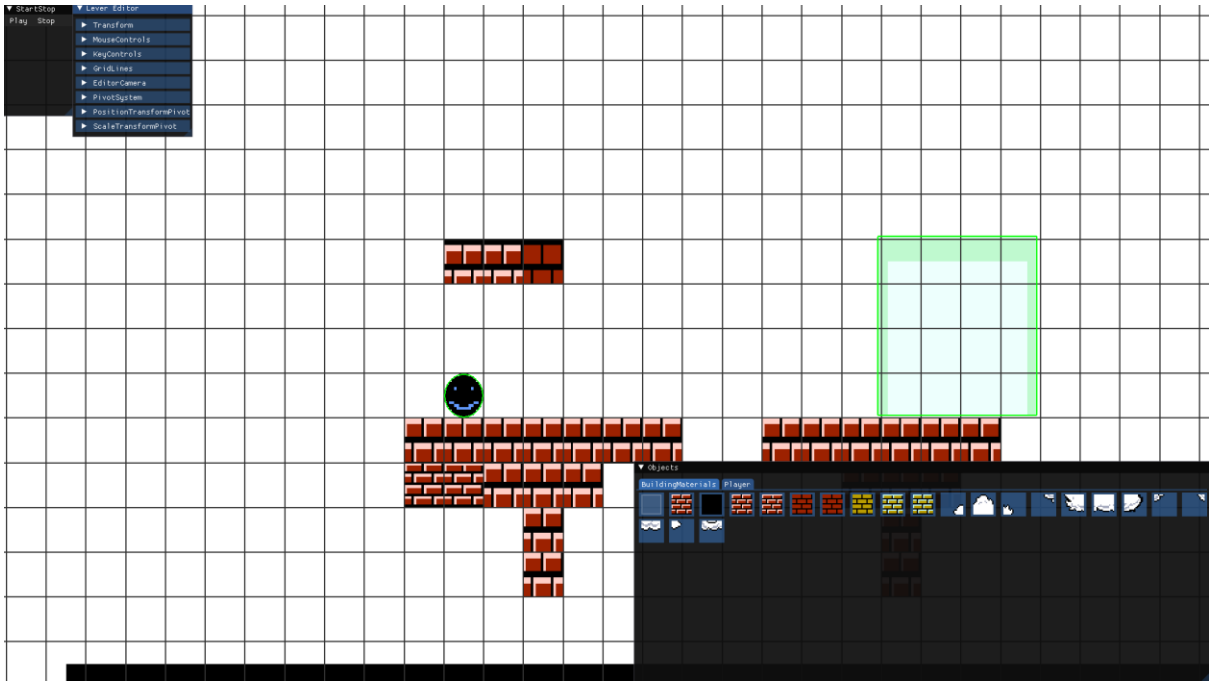
Program jsem dělal v Javě 8 za pomoci knihoven LWJGL3, GSON, Jbox2D a ImGui.

2. Grafické Rozhraní

2.1 Editor hry

Scéna, která se zobrazí hned po zapnutí programu. Hráč si zde může upravovat rozložení hry. Může klikat na jednotlivé bloky a postavit si z nich level. Má možnost blocky roztahovat (klávesa R po vybrání bloku) nebo posouvat (klávesa E po vybrání bloku) či změnit to, zda-li jsou například bloky ovlivňovány gravitací či kolik váží (prostor pro tvorbu originálních levelů).

Jednotlivé bloky si uživatel tahá z pravého spodního okna (viz. Obr. 1) a umisťuje je do pole, kde se přichitávají na křížovou síť. Uživatel může přidat hráče překliknutím na záložku player a přetažením hráče do pole.(Bohužel zatím není ošetřeno, aby nemohl být víc než jeden hráč). Přetažením prvního bloku v nabídce hráč vymezí kde je zóna pro výhru. Třetím tlačítem hráč umítí zónu pro porážku (spodní část obrázku). za V levém horním rohu má hráč možnost sputit hru, kdy okamžitě začne hrát za kuliču v poli.



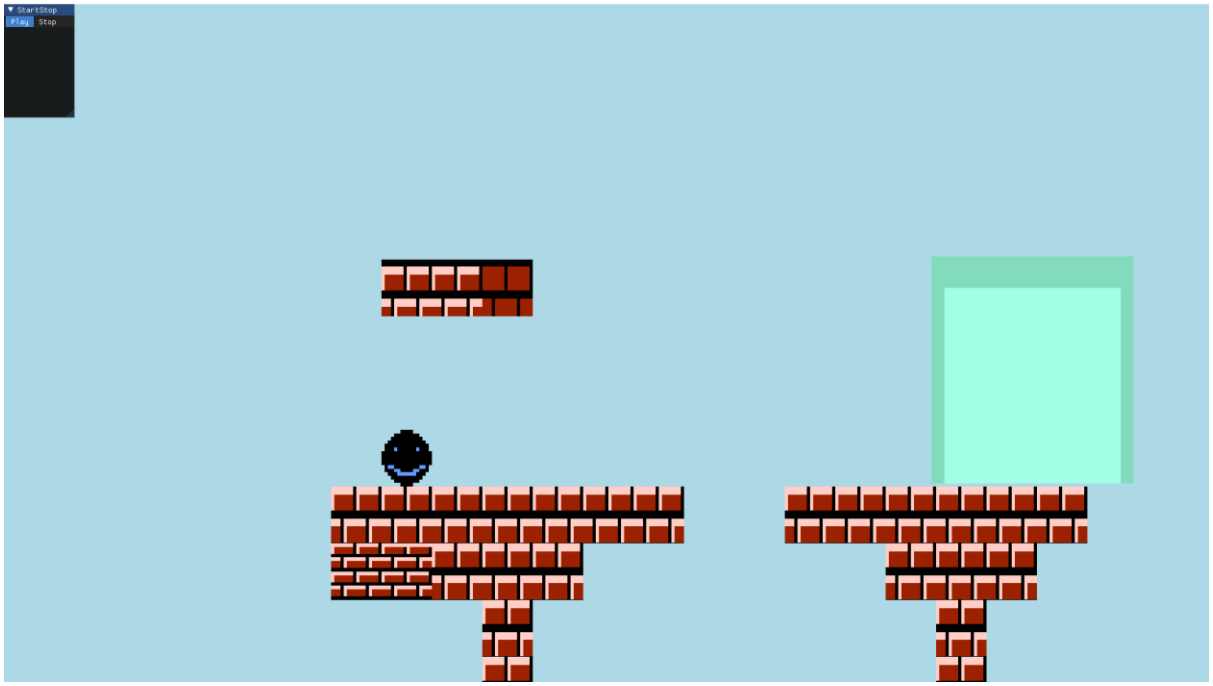
Obr. 1

2.2 Herní scéna

Jestiže hráč klikl na Play pude hned přesunut do nově vytvořené scény, ve které budou stejné objekty.
Pokud hráč dohraje hru bude vrácen do editoru, herní postava se vratí na své místo a v konzoli se vypíše, že vyhrál.
Pokud hráč umře bude také vrácen do editoru a do konzole se vypíše, že prohrál. Když hráč chce ukončit hru dříve klikne na tlačítko stop. Ve všech případech vše vypadá stejně, jako před spuštěním.

Hráče sleduje camera na kažem jeho kroku. Tedy neuteče mimo obraz.

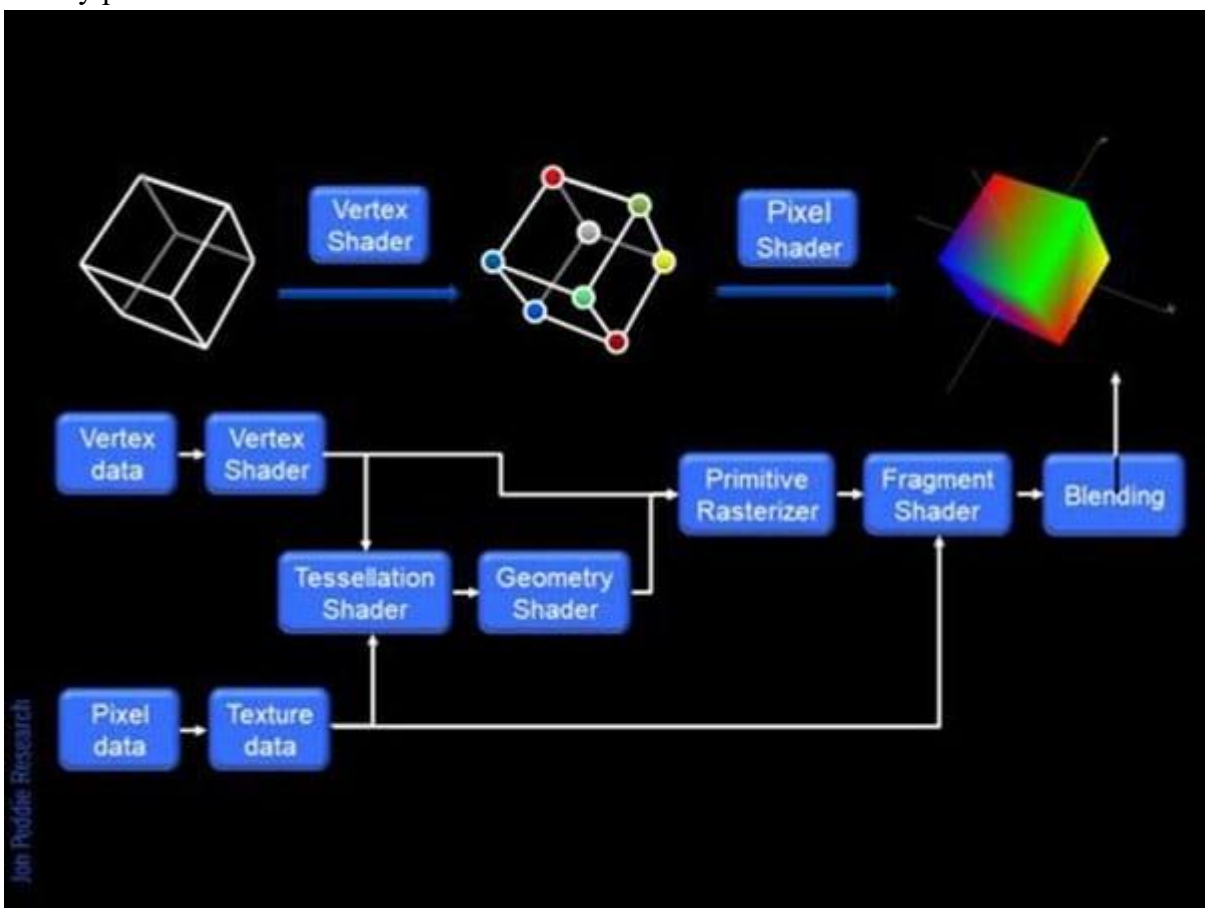
Během hry se nezobrazuje mřížka, ani ostatní panely mimo StartStop, navíc pozadí se nastaví na modré. Hráč může přidat dekorativní mráčky pro lepší dojem.



Obr. 2

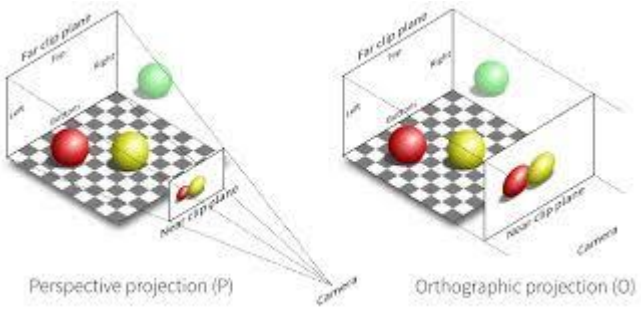
2.3 Herní Objekty

Vše na obrazovce(mimo rozhraní editoru) je herní objekt nebo komponent herního objektu. Objekty, které chceme, aby byly znázorněny graficky mají komponent SpriteRenderer, který slouží k nastavní textur na objekty. Tento komponent také umožňuje měnit barvu textur v editoru. Uchovává si sprite, což je vlastně textura. Sprite se získá ze Spritesheetu, který si uchovává obrázek s mnoha texturami v jednom obázku a přiřadí rohům hodnotu od 0 do 1. Jednotlivý Sprite dostane souřadnice v tomto rozmezí a ví, jaká část je jeho. Pro převod obrázku na texturu pro spriterenderer slouží třída AssetPool a nasledně její funkce getTexture, která převede obrázek na Texturu. Třída Texture vezme cestu k obrázku a převedeho pomoci knihovny openGL na Texturu pokud má formát RGB nebo RGBA. Třída Renderer slouží k vykreslení herních objektů. Aby se nemuseli objekty vykreslova pojednom, tak se jich vykresluje několik najednou. K tomu slouží třída RenderInBatch, do které Renderer nasype několik SpriteRendererů najednou. RenderInBatch nasypaná data předá data, opět pomocí openGL, grafické kartě, kde Shader vezme vertexi a jejich hodnoty a projekci a pohled naší kamery. Přiřadí Vertexi na místo tak aby seděli zobrazní kamery a dá jim data. Následně tyto data pošle shaderu na fragmenty a tento shader rozporstře hodnoty do prostoru. Pokud měl Vertex na shaderu přiřazenou texuru vynasobí barvu texturou. Texturu nahraje také do jednotlivich rohů. Je teda důležité správě nastavit, kam která souřadnice textury patří.



(Obr. 3 postup vykreslování (některé věci dělá glfw samo))

Kamera má zaúkol kolmím promítáním zobrazit naše objekty z kolmého pohledu.



(Obr. 4)

ImGui(Immidiete mode) má zaúkol umožnit upravování prostředí, když je program spuštění. ImGuiIntegration je třída používající kód z repozitáře ImGui.Metoda imgui je implemetována do scény a je pak volána v této třídě ImGuiIntegration kde se updatuje. Návody jak co použít lze vzít z metody ImGui.showWindow(), která ukáže všechny možné metody použití.

3. Logika aplikace

3.1 Implementace fyziky

Na propojení logické a grafické části mám třídu Physics2D, přičemž Collidery i Rigidbody rozšiřují komponenty herního objecktu a dovolují, aby měli jednotlivé objekty mezi sebou aktivovanou základní herní fyziku.

Rigidbody má 3 možné typy a to Kinematic, Static a Dynamic, přičemžna Static neplatí gravitace a je tak nastaven defaultně pro všechny stavební bloky. Dynamic má herní postava, díky čemuž na ni funguje gravitace. Třída ContactListener dovoluje řešit určité kolize. Např. hráč vejde do místa určeného pro vítězství nebo naopak prohru. Implementace fyziky pro hráče se nachází ve třídě PlayerController, kde jsem hodnoty zkoušel nastavit metodou pokus omyl, tak aby hra byla hratelná a postava neskákala na Měsíc nebo neběhala rychlostí světla.

3.2 Controller hry

Kontrolery hry jsou celkem tři přičemž jeden není uplně implemntovám, tam kdeby měl být. V editoru funguje PivotSystem, který dovoluje tahat za šipky a posout tím objekty. Klávesou R se přepne do ScaleTransformPivotu, který mění rozměry. Stiskem E se změní zpět na přesouvání objektů PositionTransformPivot. Pokud je označen objekt a stisknete klávesu DELETE objeckt se zničí a bude odtraněn ze scény. MouseControls dovolují pokládat objekty a označovat objekty. Třída EditorCamera dovoluje hýbat kamerou a stiskem a držením kolečka a tahem myši se dá hýbat kamerou. Otáčením kolečkem na myši lze přibližovat a oddalovat kameru. Stiskem T se kamera vycentruje. Hráč se ovládaá pomocí PlayerControlleru kde stistk A nebo šipky doleva pohybuje hráčem vlevo. Stisk D nebo šipky naopak pohybuje s hráčem vlevo. Stisk mezerníku, W nebo šipky nahoru zahájí skok.

4. Závěr

Myslím si, že by práce šla ještě vylepšit. Nachází se zde poměrně hodně chyb, které by chtěli ošetřit. Nadruhou stranu jsem za posledních 48 hodin zvládl kód zprovoznit. Také jsem splnil úkol co jsem si dal v druhém ročníku, aby další hra (pokud hra) se pravidelně updateovala, což jsem splnil.

I přes nechuť se prokousávat začátkem projektu, jsem začal poměrně svižně, ovšem bohužel tempo opadlo, a tak jsem měl problém projekt dokončit.

5. Zdroje

1, Jak neukládat všechny parametry do Gsonu:
https://stackoverflow.com/questions/4802887/gson-how-to-exclude-specific-fields-from-serialization-without-annotations
2. Ortographic Projection:
https://www.youtube.com/watch?v=isDMUZg0EaQ 3.JavaFX ProgressBar: how to change bar color?
3. LWJGL kód pro spusteni
https://www.lwjgl.org/guide
4. LWJGL tutorial
https://coffeebeancode.gitbook.io/lwjgl-game-design
4. GSON
https://github.com/google/gson
5. JBOX2D physics engine
https://github.com/jbox2d/jbox2d
6. BOX2D dokumentace
https://box2d.org/documentation/index.html#autotoc_md4
7. video kde jsem narazil na Box2D a port do javy
https://www.youtube.com/watch?v=MsRROjQJxuo
8. Input Listenery
https://www.glfw.org/docs/latest/input_guide.html
9. Shader tutorial
https://learnopengl.com/Getting-started/Hello-Triangle
10. Shader tutorial
https://www.youtube.com/watch?v=AjQ6U-xEDmw
11. Obrázek 4
https://www.google.com/imgres?q=orthographic%20projection%20camera&imgurl=https%3A%2F%2Fi.stack.imgur.com%2Fq1SNB.png&imgrefurl=https%3A%2F%2Fstackoverflow.com%2Fquestions%2F36573283%2Ffrom-perspective-picture-to-orthographic-picture&docid=En0sMGo22wiOcM&tbnid=oF9aZLbT5UmooM&vet=12ahUKEwib1en157GFAxVrwAIHHeCkBeUQM3oECBMQAA..i&w=1000&h=482&hcb=2&ved=2ahUKEwib1en157GFAxVrwAIHHeCkBeUQM3oECBMQAA

12. Obrázek 3
https://www.quora.com/What-is-a-shader-in-GPU

6.Obrázky