Středoškolská odborná činnost

Obor SOČ: 18. Informatika

Simplex RPG Engine

Matěj Štágl

Kraj: Liberecký

Česká Lípa 2017

Středoškolská odborná činnost

Obor SOČ: 18. Informatika

Simplex RPG Engine

Autoři: Matěj Štágl

Škola: Gymnázium Česká Lípa, Žitavská 2969, 470 06

Kraj: Liberecký

Konzultant: Mgr. Pavla Machová

V České Lípě dne 21. 3. 2017 Matěj Štágl

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v seznamu vloženém v práci SOČ.

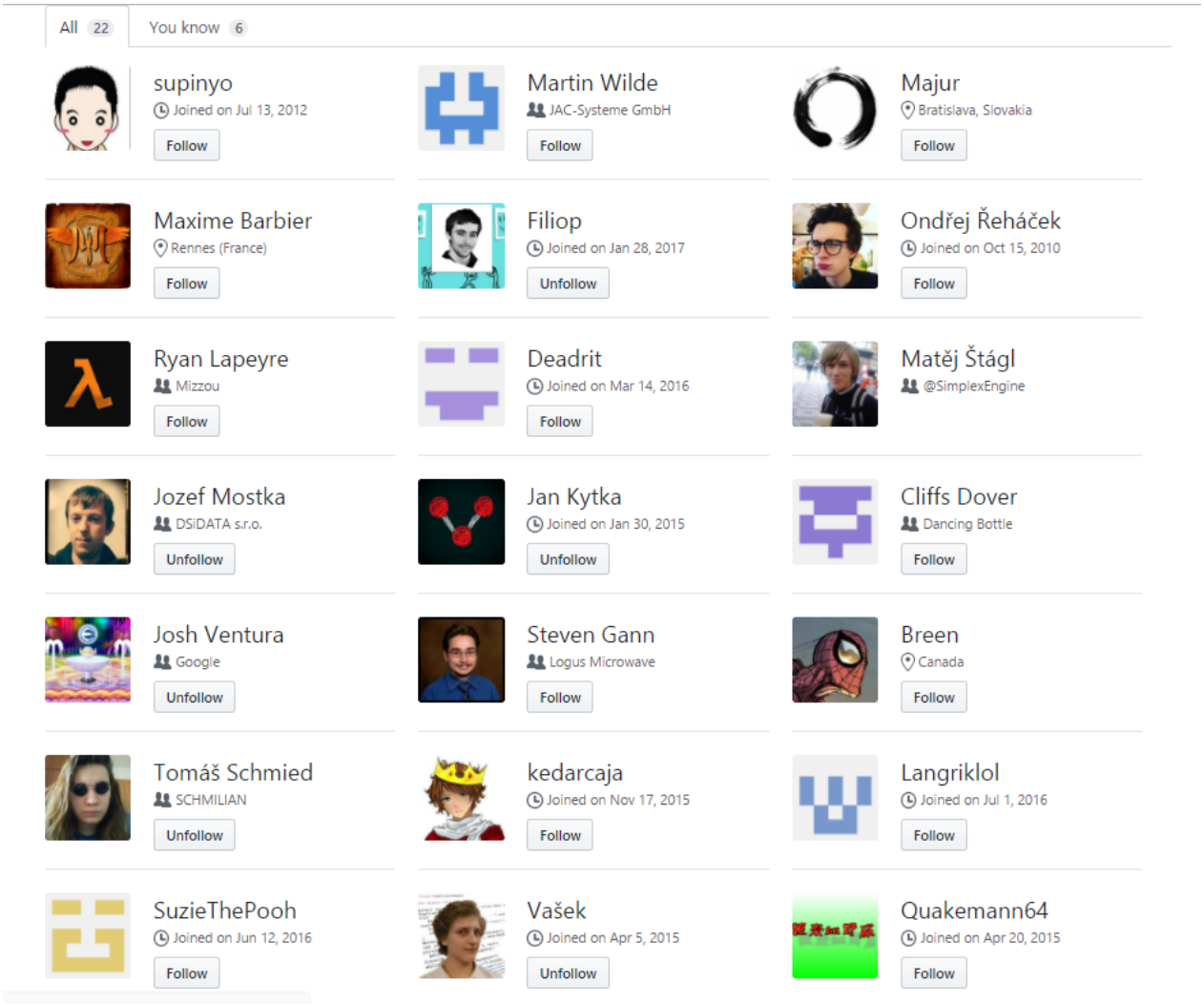
Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V České Lípě dne 21. 3. 2017 Matěj Štágl……

Poděkování

Tímto bych chtěl poděkovat všem stargazerům projektu Simplex RPG Engine. Jejich podpora je pro mě velkou motivací.



Obrázek 1: Poděkování

Anotace

Hlavním tématem práce je tvorba herního enginu, spojujícího výkonné prototypovací prostředí nástroje Game Maker: Studio a kapacity jazyka C#.

Práce seznamuje čtenáře s projektem Simplex RPG Engine a vysvětluje některé pojmy, spojené s herním vývojem.

Klíčová slova

Herní engine, C#, GML

Annotation

This work describes the process of developing a game engine, connecting powerful prototyping environment of the Game Maker: Studio and capacity of the C# language.

This work introduces the reader with the Simplex RPG Engine project and among other things it explains some terms related to the game development.

Keywords

Game engine, C#, GML

Obsah

1 Úvod

Motivací pro vytvoření specializovaného herního enginu pro mě byla absence moderního nástroje, který by byl dostatečně intuitivní pro začínající programátory a zároveň dostatečně výkonný pro firmy, produkující komerční aplikace v týmech, využívající Source control technologii a potřebující vysoce optimalizované prostředí.

Proto jsem vytvořil nástroj, sloužící jako spojovací článek mezi nástrojem Game Maker: Studio (dále označovaného jako GMS) a jazykem C#, kombinující výhody a minimalizující nedostatky tak, aby umožnil rapidní tvorbu složitých aplikací.

Odtud plyne název projektu, slovo *Simplex* je složeninou slov *simple* a *complex* – engine tedy přináší jednoduchá řešení pro složité problémy.

Simplex funguje jako nadstavba GMS, do něhož se naimportuje ve formě balíku zdrojů.

Obrázek 2: Logo projektu

2 Co je herní engine

Herní engine je nástroj abstrahující určitý programovací jazyk blíže k uživateli, poskytující řešení pro často se opakující problémy. Tím zjednodušuje proces tvorby hry, snižuje čas potřebný k jejímu vývoji a též i finanční prostředky k jejímu dokončení nezbytné.

3 Cíle projektu Simplex RPG Engine

- Přizpůsobit design GMS projektu tak, aby se s rostoucí složitostí projektu minimálně zvedal počet nových zdrojů.
- Připravit snadno integrovatelný balík komponent, řešící běžné problémy, svázané s RPG žánrem. Komponenty musí být na sobě nezávislé, mohou čerpat z jádra – tedy z jediné nezbytné komponenty, řešící ty nejobecnější problémy.
- Zavést nové struktury a standardy, usnadňující a zpřehledňující psaní kódu.
- Rozšířit prostředí GMS o externí nástroje napsané v jazyce C#, sloužící k efektivnější práci v enginu. Výstupem těchto nástrojů je typicky soubor, který se automaticky integruje do GMS projektu.
- Podpora modelu write once, compile everywhere.

4 Vývoj

Engine vznikal pod pracovním názvem *„Dragon Rise 3“*, obrázek zachycuje stav projektu v létě 2015. Prvními funkcemi byl jednoduchý model kamer, které bylo možné připnout na objekt, jež následovaly. Dále je viditelná raná verze inventáře, ovladatelná postava (animující svůj pohyb) a truhly sloužící k uskladnění sbíratelných předmětů.

Obrázek 3: Stav před commitem 1

Stav projektu o dva měsíce později, patrné jsou změny ve vykreslování podkladu, vylepšené vrstvy, rozšířená funkcionalita inventáře, první návrh komponenty equipmentu.

Obrázek 4: Stav před commitem 1 (2)

Tento obrázek zachycuje projekt po umístění jeho zdrojového kódu na síť gitHub a tím položení openSource myšlenky enginu. Patrné je vylepšené UI, nová komponenta SimpleAI, jejímž příkladem jsou objekty „slimáků“, kteří se umějí pohybovat, sesílat schopnosti a jednoduše reagovat na hráče.

Obrázek 5: Stav v okolí commitu 1

Stav projektu okolo commitu 80, obrázek zachycuje vylepšené UI, novou komponentu minimapy (vpravo dole), ukazatel zkušeností a sloty pro kouzla (vlevo od minimapy).

Obrázek 6: Stav v okolí commitu 80

V okolí commitu 120 je viditelné formování stabilizovaného UI konceptu, sestávajícího z ikon umístěných na dolní plošině, též formování ukazatele zkušeností a úrovně v dolním středu, následují sloty na kouzla, minimapa, nahoře ukazatele důležitých proměnných (život, štít, mana, stamina), vlevo aktivní okno. Taktéž byla přidána komponenta bestiáře.

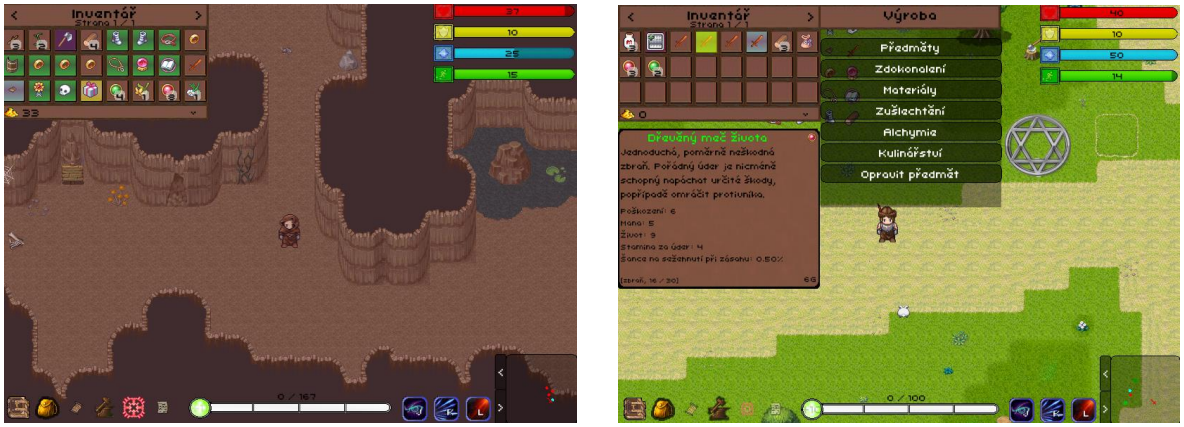
Obrázek 7: Stav v okolí commitu 120

Commit 200 ukazuje finální verzi prvotního UI designu a dokončené kriticky důležité komponenty (inventář, bestiář, úkoly, výroba předmětů, equipment, status), upravenou minimapu, novou komponentu umožňující pozměnit render výstup overall shaderem a implementaci překryvů.

Obrázek 8: Stav v okolí commitu 200

Posun v commitu 400 - patrné je razantní vylepšení inventáře, přidány byly komponenty zabezpečující počasí a střídání dne/noci, nová komponenta *BodyCanvas*, rendrující oblečení hráče

v jednotlivých vrstvách a tím umožňující snadnou vizualizaci všech nasazených předmětů.



Obrázek 9: Stav v okolí commitu 400

Stav okolo commitu 500 ukazuje nový koncept výchozího UI a některá dialogová okna – inventář, výroba předmětů, bestiář, nová flexibilní minimapa, dynamické sloty jako náhrada starých slotů na kouzla a vylepšený ukazatel zkušeností.

Obrázek 10: Stav v okolí commitu 500

Finální obrázek z vývoje zachycující projekt okolo commitu 540. Patrná je nová grafika (vytvořená na míru pro engine), nové schopnosti inventáře, paralaxní mapování.

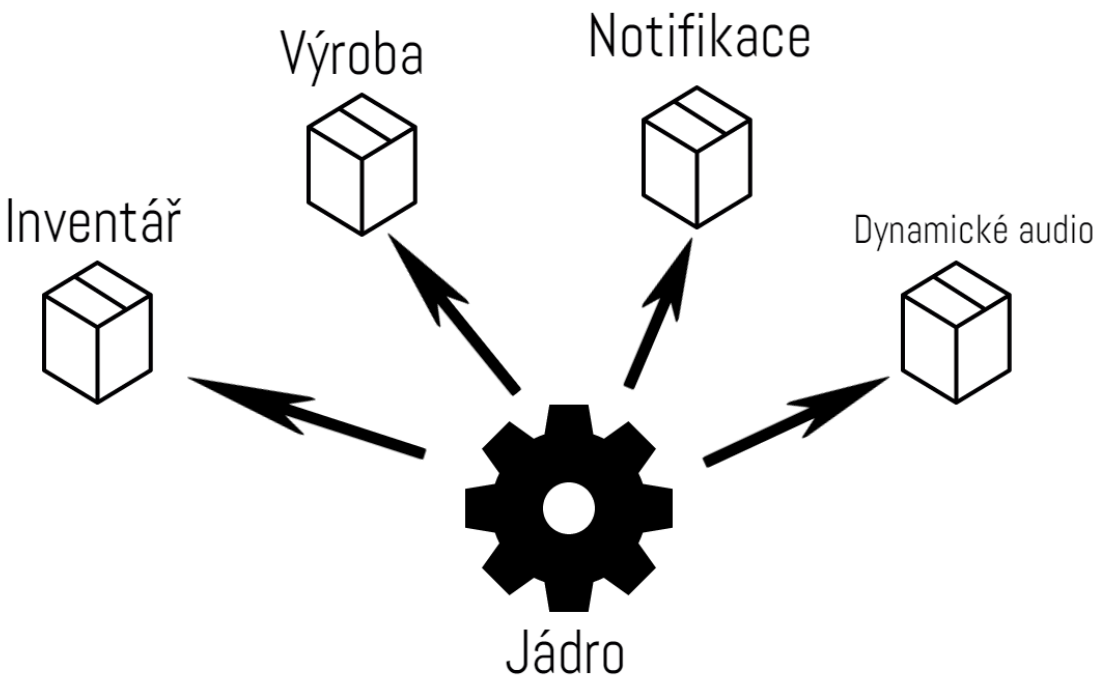
Obrázek 11: Stav v okolí commitu 540

5 Seznam funkcí enginu - úvod

Následuje stručný popis funkcí enginu, uvést ho v plném rozsahu je vzhledem k rozsahu této práce nemožné, neboť pouze připravené skripty čítají zhruba 20 000 řádků kódu, logika samotných objektů obsahuje počet řádků kódu násobně vyšší. Zmíněno je tedy pouze to nejdůležitější.

Engine funguje na principu jádra a přídavných komponent. Jádro samotné obsahuje zejména různá API, která zahrnují matematickou, render, UI a room knihovny, ale i skripty pro ovládání nejdůležitějších komponent, jako je hráč, NPC/Mob postavy. Dále obsahuje běžně využívané rozšiřující funkce, doplňující výbavu GMS, využitelnou v obecné rovině. V neposlední řadě jsou též přítomny metody pro manipulaci s poli, různé konverze, funkce zjednodušující složité, leč často se opakující bloky kódu („asm knihovna“).

Komponenty jsou vzájemně na sobě nezávislé balíky, které ke své funkcionalitě vyžadují pouze přítomnost jádra a dají se libovolně přidávat/odnímat. Řeší vždy jeden konkrétní problém sérií skriptů, které jsou poté pouze volány příslušným objektem. Typickými příklady komponent mohou být například inventář, osvětlení, dialogy nebo dynamické audio.



Obrázek 12: Náhled vztahu komponent a jádra

5.1 Seznam funkcí enginu

Jádro

API

- Matematika
- Render
- Hráč
- NPC
- Místnosti
- Nepřátelé
- Uživatelské rozhraní

Běžné

- Rozšiřující funkce k GML
- ASM knihovna
- Kritické jádro

Komponenty

- Inventář
- Truhly
- Předměty
- Equipment (formulář pro nasazení předmětů)
- Log (notifikační upozornění)
- Gore (fyzické efekty, využívané zejména k simulaci krve)
- Kolize
- Boj
- MapIntro (notifikace při vstupu do nové místnosti)
- Bestiář

- Status
- Minimapa
- Cyklus dne a noci
- Základní transformace
- Úkoly
- Cutscény
- Pauza
- Počasí
- Ovlivnění
- Zničitelné kontejnery
- Háčkování zámků
- Dráhy
- Výroba předmětů
- Portály
- Obchod
- Jednoduché osvětlení
- Výstavba v reálném čase
- Třes obrazovky
- Ocenění
- Časovače
- BodyCanvas (systém kreslení složitého objektu ve vrstvách)
- Umělá inteligence
    - Jednoduchá
    - Pokročilá

- PopUp (vyskakovací okna)
- AreaScripts (triggrovatelné události)
- Hoditelné předměty
- Ukládání a načítání
- Záblesky obrazovky
- Notifikace
- Herní svět
- Sbírka
- Aktivní model komponent
- Tvorba postavy
- Převod místnosti na bezztrátové png
- Dynamické audio
- Efekty
    - Částice
    - Shadery
    - Ostatní

7 Popis jádra a komponent

API

Matematika

Přidává zejména interpolace, implementuje 2D a 3D vektory, obsahuje i algoritmy jako vykreslení úsečky, které se využívají k trasování jednotlivých bodů.

Render

Obsahuje důležité nadstavbové algoritmy jako vykreslení části kružnice/prstenu, tečkované vykreslení základních útvarů, motion blur, vykreslení stínu nebo velmi jednoduchého osvětlení.

Hráč

Série příkazů pro hráče od zvýšení určitého atributu přes pohyb na cílovou lokaci po vynucení přechodu na nejbližší volnou lokaci. Další zajímavé funkce jsou apiPlayerSay(input) – vytvoří nad hráčem bublinu s textem, případně ji serializuje do fronty, nebo apiPlayerSplashEmoticon(index) – vytvoří emotikon, který po chvíli vybledne a ztratí se.

NPC

Skripty pro primárně přátelské postavy ovládané počítačem, obsahují say() funkce z hráčova API, dále pomocné příkazy pro získání potřebného obrázku z animace.

Místnosti

Obsluhuje přípravu místnosti, přechody mezi místnostmi, přidání NPC postavy do určité místnosti a náhradu za GMS vestavěnou funkci room_goto(room), kterou engine kvůli svému způsobu optimalizace neumožnuje přímo použít.

Nepřátelé

Stará se o instance objektů, dědících z objektu parEnemy od jejich vytvoření až po zánik. Obsahuje např. skripty pro inicializaci, absorpci poškození, reakci na útok, vypadnutí požadovaných předmětů po smrti, say() rodinu a aplikování ovlivnění.

Uživatelské rozhraní

Obsahuje nejběžnější skripty pro tvorbu UI, ty se využívají jednak ve výchozím layoutu, ale jsou určené i pro návrh vlastního designu.

Běžné

Rozšiřující funkce k GML

Obsahuje pomocné funkce, které zjednodušují velmi běžné úlohy. Například kontrolu, zda se kurzor nachází v zadaném čtverci, n-tou nejbližší instanci od dané pozice, rozšiřující funkce pro práci s textovými řetězci.

ASM knihovna

Balík funkcí, které slouží k jednoduššímu a zároveň kombinovanému volání funkcí draw_set_ / font_set_ / audio_play_ / image_x/y_scale. Jsou nazvány vždy třemi písmeny (odtud název, podobnost s jazykem symbolických adres), např. clr(), alg(), sfx(), scl().

Kritické jádro

Obsahuje definici systému SimplexString – textového řetězce, obsahujícího značky, který se zpětně parsuje, snadnou deklaraci/uvolnění částicových systémů, konverzi indexů polí (2D -> 1D), skloňování, kostku, zalamování textu a další.

Komponenty

Inventář

Jeden z velmi typických prvků pro RPG žánr. Inventář umožnuje plně virtuální (bezinstancové) zobrazování předmětů, vzácnost, statické a dynamické atributy předmětů, možnost označit předměty jako oblíbené, či jako odpad, kontextové nabídky pro každý předmět, stackovatelnost, kombinování předmětů, infobox, trvanlivost, výpočet reálných atributů na základě poškození, setřídění předmětů, vyhledávácí filtry, identifikaci, informační text rozdělený na hlavičku – tělo – patičku.

Truhly

Umožnují nastavení výchozích předmětů, počtu slotů a řádků v creation kódu instance, odložení/vyjmutí předmětu z truhly, zamykatelnost (v takovém případě nutnost vlastnit klíč/vyháčkovat zámek).

Předměty

Velmi rozsáhlá kolekce skriptů pro definici atributů předmětu, když dojde k jeho vytvoření, nastavení kontextových možností, vzácnosti, obsluhy vykreslovacího módu a mnoho dalšího.

Equipment

Skripty pro nasazení či sundání předmětů buď přetažením, nebo rychlou událostí, automatické porovnávání předmětů, poškození vybavení a vykreslení obslužného UI.

Log

Komponenta se širokým využitím, od zobrazení poškození při zásahu po zprávy s 3D efekty při sebrání předmětu. Obsahuje jediný skript scrLog(), který lze pomocí parametrů velmi přizpůsobit.

Gore

Obsahuje sérii skriptů pro efekty simulující krev se stupňující se intenzitou, taktéž rozložení bitmapy na libovolně velké části a jejich následnou explozi.

Kolize

Skripty zabezpečující inicializaci kolizních objektů, které umožnují velmi optimalizovanou a zároveň libovolně detailní interakci s prostředím.

Boj

Implementace bojového systému, který zajišťuje, že objekt nemůže být vícekrát ovlivněn jedním útokem, dále obsahuje skripty pro zjištění, zda byl provedený útok kritický či zablokovaný a obsluhuje pomocný objekt, který zajišťuje potřebné kolizní zóny.

MapIntro

Jednoduchá komponenta, obsahující pouze skript scrMapIntro(), který vytvoří efektní animaci při vstupu do nové místnosti.

Bestiář

Obsahuje skripty pro inicializaci a vykreslení bestiáře. Ten umí paginaci, náhledy a detaily jednotlivých subjektů, získávání postupných informací a fragmentů, garantující bonusy v boji.

State

Komponenta zaznamenávající důležité události, které řadí do fronty a následně vykresluje na obrazovce. Kromě inicializačního a kreslícího skriptu obsahuje metodu stateAddEntry(), která zařadí do fronty určený text, ten může obsahovat barevné značky nebo je obarven celý určenou barvou jako nepovinný parametr.

Minimapa

Důležité jsou skripty scrMinimapAddWaypoint() a scrMinimapRemoveWaypoint(), které přidají či odeberou z mapy důležitý bod, ke kterému pak minimapa umí sama navigovat. Tento bod může být označen textem nebo bitmapou a může obsahovat text, zobrazující se při podržení kurzoru nad ním.

Střídání dne a noci

Sebepopisující komponenta, skript daynightFlow() nechá téct čas o určitý počet sekund, noc výrazně omezuje výhled hráče, který mohou rozšířit objekty emitující světlo.

Základní transformace

Obsahuje skripty aplikovatelné na všechny potomky objektu parSolid, například metodu scrBasicTransformScale(), která jako parametry bere cílové x/y měřítko. O interpolaci se postará samotný systém.

Úkoly

Bohatá sbírka skriptů pro zadávání / manipulaci / odebírání úkolů. Úkoly mohou postupně získávat dodatečné poznámky o postupu pomocí skriptu scrQuestEntrySetText(), pokud se nepovinný parametr „addAsNote“ nastaví na hodnotu true.

Pauza

Komponenta zabezpečující pozastavení hry a vykreslení menu, obsahuje nastavení a různé nabídky.

Počasí

Umožnuje v reálném čase generovat efekty jako déšť, sníh a podobné.

Ovlivnění

Obsluhuje životní cyklus „ovlivnění“ umístěných na hráče, tedy efektů jak kladných, tak i záporných, například „prokletí, regenerační aura, sytost, otrava jedem“. Ovlivnění v čase ztrácejí životnost a po vypršení tohoto času zaniknou, parametrem je možné nastavit, zda má při umístění nového stacku dojít k nastavení životnosti na výchozí úroveň.

Zničitelné kontejnery

Obsahuje důležitý skript scrBCAddLoot(), který přidává do kontejneru poklad, jenž vypadne na zem po zničení kontejneru.

Háčkování zámků

Řada skriptů pro nastavení počtu stavítek v zámku, správného pořadí jejich uvolnění a složitosti zámku.

Dráhy

Komponenta sloužící k vizualizaci pohybu, obsahuje i funkce k pokročilé modulaci dráhy jako rozptyl a drift. Vnitřně funguje na bázi vertexového vykreslování.

Výroba předmětů

Rozsáhlá sada skriptů umožňující modifikaci předmětů, obsahuje možnosti pro výrobu předmětu, jeho úpravu (zaseknutí jiného předmětu či vylepšení), zušlechtění materiálů, alchymii, kulinářství, stavění struktur.

Portály

Minimalistická komponenta umožňující přesun hráče z místa A do místa B. Parametrem lze zajistit, že bude přesunuta i kamera, nebo k novému místu interpoluje.

Obchod

Nadstavba pro NPC postavy, obsahuje funkce pro přidání nových položek do obchodu, nastavení dočasných slev, systém poptávky a nabídky, prodej předmětů.

Jednoduché osvětlení

Na výpočetní čas nenáročná komponenta, poskytuje systém, který je snadno integrovatelný do stávajícího projektu, světla se nicméně nelámou o objekty, ale volně prostupují dál.

Výstavba v reálném čase

Umožňuje budovat stavby po jednotlivých dílech, přičemž ty se logicky spojují a mění svůj vzhled tak, aby vypadaly plynule.

Třesení obrazovky

Další komponenta z rodiny vizuálních efektů, nepovinným parametrem ve skriptu scrScreenShake() lze nastavit, zda se bude třást i UI, nebo pouze podklad.

Ocenění

Fungují na principu registrace ocenění v creatu pod identifikátorem a při vhodné události je ocenění pouze odemknuto. Generuje vizuální efekt, který je používán i pro notifikaci při novém či splněném úkolu.

Časovače

Slouží jako náhrada za výchozí GM „alarmy“, oproti nim disponují možností dočasného pozastavení, navíc jejich počet není nijak limitován (u GM je max. počet 12).

BodyCanvas

Komponenta, která je schopná vykreslit objekt po jednotlivých částech, jež lze libovolně měnit, primárně je využívána k simulaci oblečení.

Umělá inteligence

Komplexní komponenta obsahující jednoduchou a složitou UI, kterou lze využívat podle dostupných výpočetních zdrojů a náročnosti situace. Jednoduchá UI nevypočítává cestu předem, ale reaguje na okolní podměty. Složitá UI předem hledá optimální cestu k cíli pomocí A* algoritmu, cestu přepočítá po fixním počtu kroků hry a při událostech. Akce vybírá na systému priorit, kdy dochází ke kontrole životně nezbytných funkcí, následně pokusu o jejich zachování a poté k méně důležitým akcím. UI označená jako vládce může též ovládnout okolní UI a používat je jako roj, jehož výpočetní kapacitou disponuje a následně zpětně udílí rozkazy.

PopUp

Náhrada za vyskakovací okna, obsahuje substituce funkcí show_message, show_question a get_string/int. Na rozdíl od výchozích GM funkcí nepozastavuje běh runneru, což je považováno za riziko v životním cyklu aplikace.

Oblastní události

Slouží k vymezení oblastí, které můžou být určitým způsobem aktivovány, což provede nastavený skript. Přímočará implementace, která je maximálně jednoduchá k použití.

Vrhatelné předměty

Minimalistická komponenta umožňující hodit předmět zvolenou silou, pokud je označen příslušnou vlajkou.

Ukládání a načítání

Simplex RPG engine hojně využívá generické struktury, a jelikož je GMS výchozími funkcemi neukládá, vznikly náhrady funkcí game_save / load. Fungují tak, že externě uloží či načtou generické struktury a následně vykonají klasický save nebo load.

Záblesk obrazovky

Vizuální efekt, který interpolací překryje obrazovku a může následně vypsat zvolený text. Interpolace může mít tvar libovolného útvaru, který se rychle zvětšuje.

Notifikace

Derivát komponenty Ocenění, přejímá vizuální reprezentaci a umožňuje vypsat libovolný text a bitmapu.

Herní svět

Obsahuje skripty pro manipulaci s předem umístěnými spawnery. Umožňuje generovat cutscény a moby.

Sbírka

Komponenta tvořená skripty pro definici a odemčení sbíratelných předmětů, podobný model jako Ocenění.

Aktivní komponenty

Systém zabraňující překrývání oken hráčem, funkcionality je dosaženo udržováním informace o „focusu“ komponenty.

Tvorba postavy

Skript scrCharacterPsychologicalEvaluation() provede psychologickou evaluaci hráče a na základě získaných informací distribuuje body atributů.

Map to png

Exportuje celou místnost jako .png soubor. Výsledkem skriptu scrMapToPng() jsou data, která musí spojit externí nástroj, jenž je součástí C# toolkitu enginu.

Dynamické audio

Komponenta umožňující střídat hrající hudbu podle událostí ve hře a připnout zvuk na objekt, čímž dochází k jeho 3D projekci.

Efekty

Rozsáhlá sbírka převážně částicových efektů, dále ovládací skripty pro GLSL shadery.

8 Shadery

Engine disponuje řadou předpřipravených GLSL shaderů, které modifikují buď výsledný obraz, který se má vykreslit, nebo pouze výstup z určitého objektu. Následuje stručný výčet a popis těch nejdůležitějších.

Color

Násobí všechny barevné kanály vstupní barvou, slouží k jednoduchému přebarvení.

Lerp

Lineárně interpoluje barevné kanály na určitou hodnotu.

Gauss

Rozostřuje obraz Gaussovou metodou rozostření.

Sepia

Převádí obraz do žluto-šedé škály.

Invert

Převrací barvy.

Thermal

Simuluje funkci termovizoru.

Sketch

Převede obraz na skicu.

Greyscale

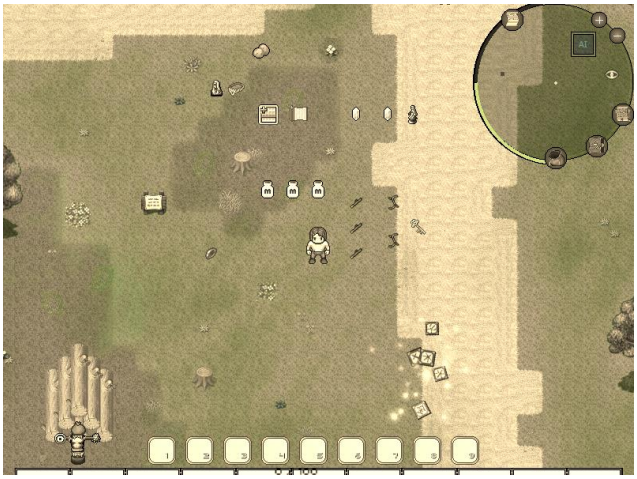
Převádí obraz do odstínů šedé.

Shockwave

Deformuje obraz simulací tlakové vlny.

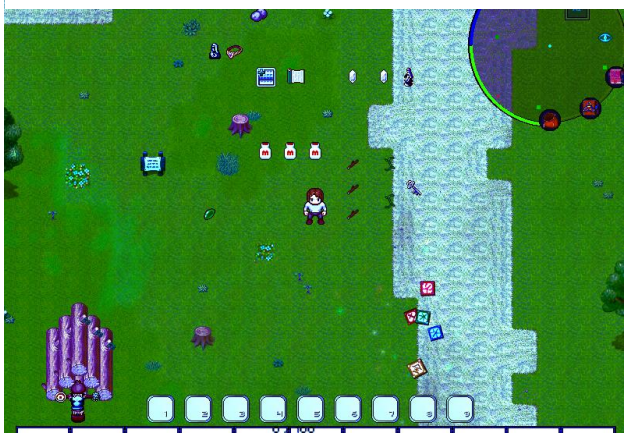
Perlin noise

Generuje měkký šum, podle kterého je vypočítávána dynamická voda.



Obrázek 14: Sepia

Obrázek 16: Invert



Obrázek 13: Lerp

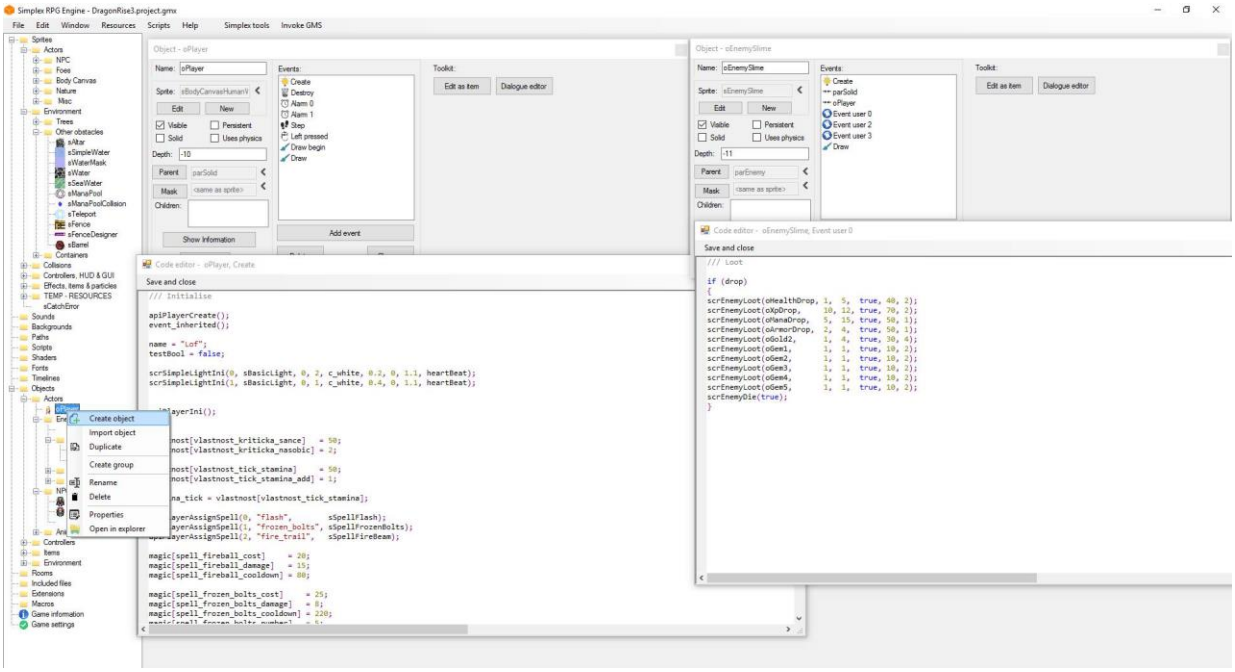
Obrázek 15: Sketch

Obrázek 17: Icy, oil

9 Externí toolkit

Engine je vybaven sadou externích nástrojů, které zvyšují efektivitu práce. Patří sem hlavní IDE, editor dialogů, cutscén, generátor textů a další. Následuje jejich popis.

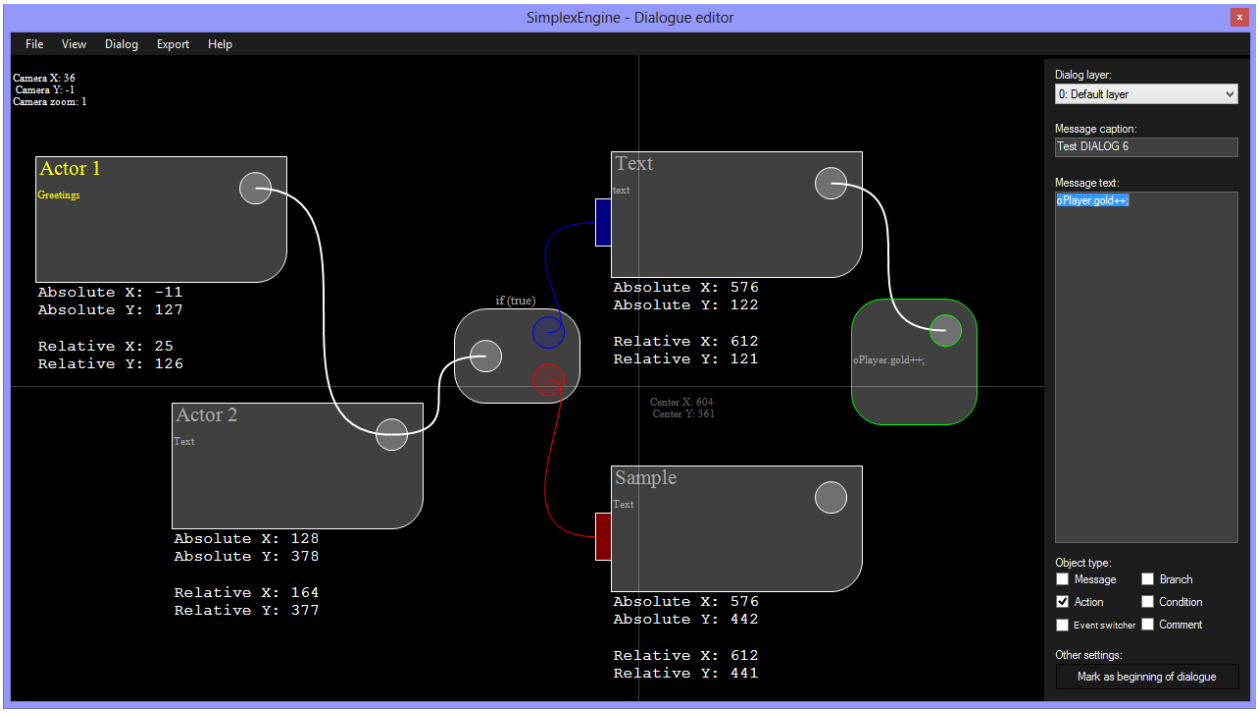
Simplex IDE



Obrázek 18: Simplex IDE

Alternativní vývojové prostředí k GMS, umožnuje načíst libovolný GMS projekt, editovat objekty a upravovat jejich události. Změny jsou ukládány zpět, po probliknutí okna GMS jsou načteny v reálném čase. Výhodou oproti GMS je velice rychlé načítaní projektů (cca 10x rychlejší než GMS), nástroj také slouží k snadnému ovládání ostatních nástrojů enginu, obsahuje editor herních ocenění. Herní sprity se dají z editoru otevřít pomocí nástroje Malování. Simplex IDE neumí projekty kompilovat, ale pokud je pod ním otevřený příslušný projekt v GMS, může být pomocí kliknutí na tlačítko *Invoke GMS* předán signál k zahájení kompilování. Simplex IDE je možno využít nezávisle na enginu k prohlížení zdrojových kódů GMS projektů a jejich editaci, pokud uživatel není vlastníkem licence pro GMS. Simplex IDE v momentálním stavu neobsahuje editor místností a nenačítá některé zdroje.

Dialog editor

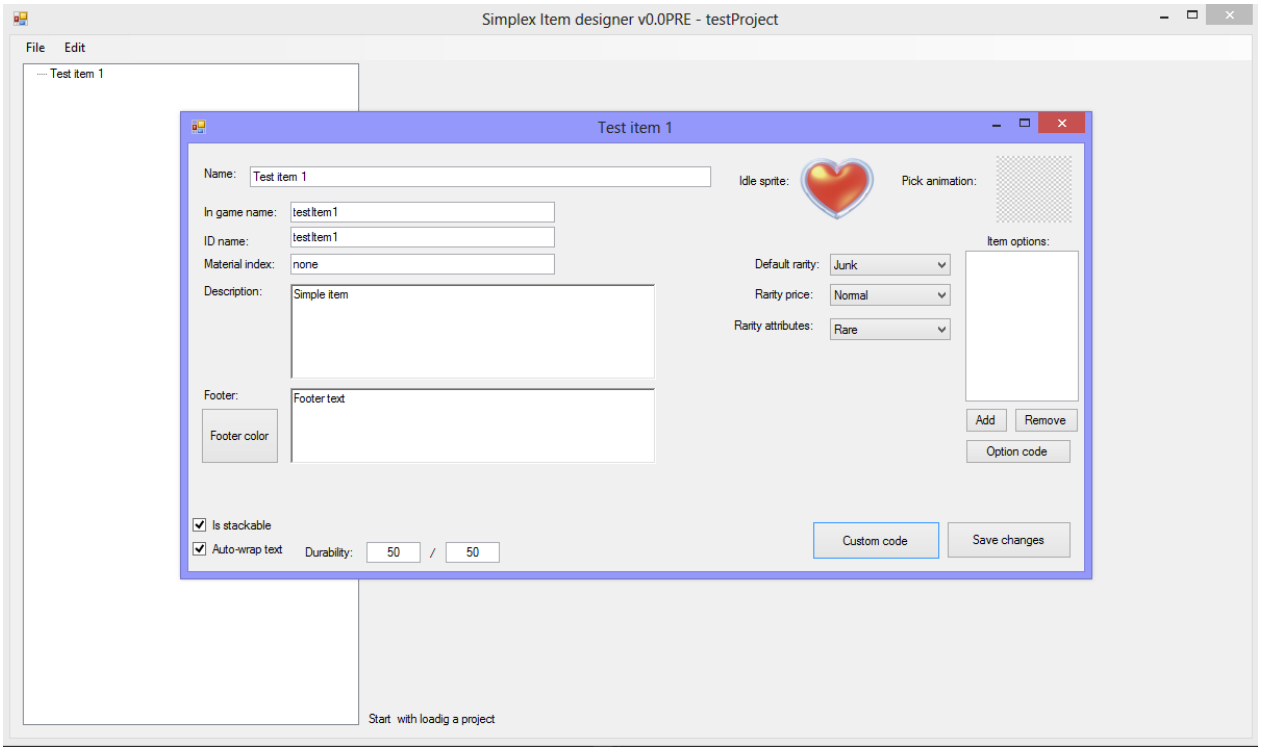


Obrázek 19: Dialog editor

Nástroj sloužící k snadnému generování dialogů mezi postavami, vizuální reprezentace je založena na modelu entit, propojení řídí tok dialogu. Obsahuje podporu vrstev dialogu, několik typů entit (zpráva, podmínka, přepínač, komentář, změna vrstvy, skript), neomezené plátno, kameru. Data jsou exportována jako GML kód připravený k použití a ukládána v XML formátu, přičemž projekt je možné zpětně načíst a dále upravovat.

Obrázek 20: Ukázka dialogu za běhu hry

Editor předmětů



Obrázek 21: Editor předmětů

Návrhář předmětů pro jejich zjednodušenou definici, generuje GMX objekt, připravený k použití. Obsahuje standardní informace o předmětu jako jméno, herní ID, index materiálu předmětu, popis (hlavička, tělo, patička), obrázek předmětu, trvanlivost, informaci o stackovatelnosti předmětu a další. Nástroj je v produkci a některé funkce nejsou v přiložené verzi implementované či stabilní.

```
1  /// Initialize item
2
3  scrItemSetUp(0, 0, itemEnum.itemHelmetHat);
4  scrItemSetProperties(false, "Štramácký klobouk", "Nedbalá elegance, kterou vyzařuje spolu s úsměvy jeho předchozích
5  scrItemSetOptions();
6
7  scrItemSetProperty(vlastnost_max_zivot, 2);
8  scrItemSetProperty(vlastnost_max_stamina, 3);
9  scrItemSetProperty(vlastnost_elegance, 1);
10 scrItemSetBodyCanvas(animationSlotsEnum.head, sBodyCanvasHead1, sBodyCanvasAttackHead2, sBodyCanvasFireHead2, sBody
11 scrItemSetUpFinalize();
12
```

Obrázek 22: Editor předmětů – generovaný kód

Ostatní nástroje

- **Návrhář barevných textů**

  Generuje text s barevnými značkami, barvy lze editovat v reálném čase.

- **RoomToPng utilita**

  Spojuje data exportovaná příslušnou komponentou enginu v jediný .png soubor.

- **Generátor cutscén**

  Generuje .gmx timeline z dat exportovaných interním designérem cutscén v enginu.

10 Význačná řešení enginu v oblasti problematiky žánru RPG

Systém interaktivního terénu

Interakce postav s okolním světem je ve větší či menší míře potřebná v každé RPG hře. A to od základních kolizí po složité chování detailních objektů.

Model použitý pro engine umožňuje paměťově efektivní zpracování kolizí s libovolnou úrovní detailu. Principem je oddělení logiky a vykreslení objektu, který má podporovat interakci. Následující obrázek ukazuje pohled na ukázkovou místnost z pohledu návrháře.

Obrázek 23: Interaktivní terén 1

Obrázek 24: Interaktivní terén 2

Následuje tentýž pohled po vypnutí vykreslení objektů:

Patrné je, že grafická reprezentace rostlin, kamenů a podobných entit zůstala vykreslena, ale jejich překryv (červený / zelený / žlutý čtverec) je skrytý.

Toto je principem fungování modelu interaktivního terénu. Grafickou reprezentací potřebných entit jsou dlaždice (tiles) – na vykreslení nenáročná bitmapa, popř. její část. Tuto dlaždici obsluhuje logický objekt – kolizní maska, ta se nevykresluje, a proto je možné ji zcela vypnout, pokud není momentálně používána, aniž by došlo ke ztrátě grafické reprezentace entity. Kolizní masky se dělí na následující druhy:

Pro postavy neprůchozí:

- **Jednoduchá solidní kolize**

  Neobsahuje žádné chování a reakce na události.

- **Pokročilá solidní kolize**

  Potomek jednoduché varianty, rozšířený o seznam zpráv, které se zobrazí při interakci s hráčem, index obsluhované dlaždice, identifikátor, vlajky pro určení různých vlastností (zničitelný objekt, počet životů, efekt při poškození atp.)

- **Přechodová kolize**

  Slouží k přechodu do nové místnosti opuštěním aktuální místnosti. Obsahuje informaci o cílové místnosti a módu umístění pro výpočet nových souřadnic postavy.

- **Lokální přechodová kolize**

  Stará se o přechod z jedné místnosti do druhé vstupem do grafické reprezentace dveří či podobné entity. Obsahuje vlajky, které mohou určit, zda je přechod uzamčený, potřebný předmět k odemčení a další.

Pro postavy průchozí

- **Nesolidní kolize**

  Obdoba pokročilé solidní kolize, obsahuje speciální vlajky určené pro průchozí kolizní objekt.

Speciální

- **Prostorová kolize**

  Řeší problém 2,5D perspektivy, markantní je tento problém u interiérových oblastí, například domů, katakomb. Podle výšky postavy a své lokace se chová jako průchozí nebo neprůchozí, neobsahuje žádnou jinou logiku.

Ukázka interakce kolizní masky ve hře:

Obrázek 25: Interaktivní terén 3

Vývoj za běhu aplikace

Engine umožňuje přepnout za běhu hry do vývojového režimu, který obsahuje nástroje pro intuitivní vytváření nového obsahu. Následující obrázek zachycuje režim tvorby cesty, podle které bude následně externím nástrojem vygenerován timeline objekt.

Obrázek 26: Vývoj za běhu aplikace

Dále je možné uzamknout kameru na libovolný objekt, popř. se pohybovat v režimu volné kamery a přesouvat objekty.

Optimalizace

Základní aplikovanou myšlenkou je v každém kroku hry zpracovat pouze ty objekty, které jsou pro hráče viditelné, a všechny objekty ze seznamu výjimek – různé kontroléry, UI a pomocné objekty. Dalším krokem je deaktivace všech kolizních masek a jejich následná aktivace pouze v okolí objektů, které je potřebují k interakci. Velkým výkonnostním bonusem je možnost přeložit projekt do jazyka C++ pomocí kompilerů YYC / Enigma a následně ho zkompilovat[1].

Obrázek 27: Vizualizace optimalizace

Obrázek zvýrazňuje optimalizaci za běhu aplikace, z kolizních objektů jsou momentálně aktivní pouze klády (červeně zvýrazněno) a pařez (žlutě zvýrazněno). O aktivaci klád si řekl objekt slimáka (vlevo nahoře), pařez aktivoval hráč, protože se nachází v jeho akčním radiu.

11 Úspěchy enginu

Simplex RPG Engine je duálně licencovaný jako placený software nebo je dostupný pod GPL3 licencí. Vzniká na něm řada projektů od nezávislých vývojářů i herních studií. Následují dva příklady projektů, které jsou již úspěšné.

Fate of hero – Kedarsoft

Úspěšná crowfundingová kampaň na síti Startovač.cz, vybráno 10 000 českých korun.

Obrázek 28: Fate of hero

Shuups! – K&P Games

Uveřejněno na platformě Google play.

Obrázek 29: Shuups!

12 Závěr

Cílem této práce bylo seznámit čtenáře s kapacitami projektu Simplex PRG Engine, nahlédnout některé principy jeho vnitřního fungování a ozřejmit ideu a důvod vzniku tohoto nástroje.

Grafiku pro engine vytvořil *Filip Ajskner*. Dále je použita veřejně dostupná grafika z projektu Liberated pixel cup a některé zdroje pod MIT / CC3 licencí.

Autorem zvukových efektů a hudby, pokud není uvedeno jinak, je *Otto Halmén*.

Zdrojové kódy projektu jsou nedílnou součástí práce a jsou v aktuálním stavu dostupné z repositáře projektu na adrese: https://github.com/lofcz/SimplexRpgEngine

Do otevřeného zdrojového kódu projektu přidalo minoritní změny několik uživatelů sítě gitHUB – Filip Ajskner (grafika), Václav Valenta (změny pro starší verzi enginu, nejsou již používány), Tomáš Schmiedt (změny pro starší verzi enginu, nejsou již používány). Celková suma řádků kódu od těchto přispěvatelů tvoří cca 3,4% celku (15 500 / 475 000).

Engine je kompatibilní s poslední stabilní verzí GMS (1.4.1763), v době vzniku této práce se program GMS2 nachází na přelomu verze beta / veřejná alfa a jeho podpora je prioritou číslo jedna ve vývoji Simplexu.

Aktuálně se engine nachází ve verzi 1.0RC6 – šestý kandidát na první stabilní release. První stabilní verze nebude obsahovat žádné zásadní novinky, pouze ladí stabilitu a doplňuje stávající funkcionalitu.

Některými alternativami k tomuto nástroji, vhodnými ke zmínění, jsou: Godot, Construct2, GDevelop a Unity.

Použitá literatura

1 Game maker: Studio - dokumentace. *Game maker: Studio* [online]. [cit. 2017-03-28]. Dostupné z: https://docs.yoyogames.com/

2 Průvodce programováním v C#. *Microsoft* [online]. [cit. 2017-03-28]. Dostupné z: https://msdn.microsoft.com/cs-cz/library/67ef8sbd.aspx

3 C#. *Wikipedie* [online]. [cit. 2017-03-28]. Dostupné z: https://cs.wikipedia.org/wiki/C_Sharp

4 Game engine. *Wikipedia* [online]. [cit. 2017-03-28]. Dostupné z: https://en.wikipedia.org/wiki/Game_engine

5 Engine definition. *Technopedia* [online]. [cit. 2017-03-28]. Dostupné z: https://www.techopedia.com/definition/24155/engine

Seznam obrázků