



TauCetiF2 – budovatelská počítačová hra se strategickými prvky

autor: Pavel Halbich | vedoucí: Mgr. Pavel Ježek, Ph.D. | 2017

TCF₂



Úvod

V současné době jsou velice populární hry s otevřeným světem. Lákají hráče na obsáhlost světa a možnost nelineárního řešení problémů a a herních úkolů. Her s otevřeným světem najdeme například množství v různých herních žánrech. My jsme se zaměřili na podmnožinu her, které kromě otevřeného světa nabízí také možnosti budování struktur a vyžadují od hráče netriviální styl hraní, který mu umožňuje ve hře přežít. V herním průmyslu se tyto hry často označují jako *sandboxové*, s *budováním*, s *průzkumem prostředí*, o *přežití*. V této práci pak představujeme naši vizuální možnosti rozvoje her tohoto žánru.

Současný stav

Provedli jsme obsáhlou rešerší a zjistili jsme, že součesné hry, jako třeba *Minecraft* nebo *Space Engineers*, mají společnou vlastnost - hráč může stavět pouze z bloků pevně dané velikosti. Zde vidíme prostor pro zlepšení. Dále jsme zjistili, že hráčům inventář vyžaduje manuální správu, což je s naší vizuální neslušitelné.

Cíle

V rámci práce jsme identifikovali následující cíle práce:

1. Bloky a herní svět
 - Způsob řešení proměnlivé velikosti bloků
 - Umístění bloků v herním světě
 - Skládání bloků do struktur
 - Komunikace bloků
 - Interakce hráče s bloky
 - Denní cyklus
 - Proměnlivé počasí
 - Efektivní datové struktury pro bloky a herní svět
2. Inventář
 - Automatizovaná správa inventáře
 - Stavitelné a umístitelné bloky
3. Herní postava
 - Inventář herní postavy
 - Vlastnosti postavy jako např. zdraví
4. Ostatní herní prvky
 - Systém ukládání a načítání hry
 - Kreativní herní módy
 - Herní tutoriál
5. Případné prvky navíc, které se ve hrách běžně vyskytují
 - Lokalizace hry do různých jazyků
 - Hudba ve hře
6. Zhodnocení práce
 - Získat a zhodnotit zpětnou vazbu (dotazník)

Nástroje

Hlavním nástrojem, který jsme při našem vývoji použili, byl *Unreal Engine*. Díky této volbě jsme se nemuseli navíc zbývat vlastní implementací herního enginu a nemuseli jsme řešit tvorbu standardních mechanik a nástrojů, které jsme v naší práci využili (například nástroj pro tvorbu *stromů chování* pro implementaci umělé inteligence).

Dalším nástrojem, který jsme využili, byl 3D modelovací nástroj *Cinema4D*, kde jsme využili možnosti tvorby 3D modelů, které jsme následně importovali do *Unreal Engine*. V souvislosti s grafikou jsme také využili grafického nástroje *Gimp*.

Architektura

V naší práci jsme využili obou způsobů vývoje počítačových her, které nám *Unreal Engine* nabízí. To jest možnosti implementace hry klasickým způsobem, tedy v jazyce C++, a dále pak možnosti implementace pomocí *Blueprintů*, což je vizuální editor kódu pro *Unreal Engine*. Využili jsme výhod obou přístupů a jejich vhodnou kombinací jsme dosáhli rychlého a efektivního vývoje celé hry.

Podařilo se nám dosáhnout plné spolupráce mezi kódem v C++ a v Blueprintu. Jeden způsob byl triviální - Blueprinty mohou dědit z C++ tříd a tudíž mohou volat metody svých předků. Z opačné strany jsme zvolili použít události, které byly vyvolávány ze strany C++ kódu a jejichž obsluha byla definována v Blueprintovém kódu. Tímto způsobem jsme dosáhli oboustranné komunikace mezi všechny součástmi hry.

Dále jsme využili komponentový přístup. Kupříkladu implementaci *kyslíkové* či *elektrické* komponenty jsme mohli tyto komponenty přiřadit jak k blokům, tak třeba k hráčově postavě. Tímto způsobem jsme se vyhnuli duplicitnímu kódů a mohli jsme s těmito mechanikami pracovat standardizovaným způsobem.

Dále byl ve hře systém bloků implementován takovým způsobem, že je možné rozšiřovat dostupné bloky pomocí *DLC*, tedy stahovatelného obsahu. Hra tedy nabízí alespoň základní možnosti uživatelkého rozšiřování.

Během práce na TCF2 jsme se maximálně snažili využít možností, které nám *Unreal Engine* nabízí. Proto jsme byli kupříkladu schopni poměrně rychle implementovat logiku, která ovládá systém počasí. Ten byl implementován pomocí rozhodovacího stromu, k čemuž jsme využili již existující nástroje, kterými *Unreal Engine* disponuje.

C++

Vzhledem k rychlosti, kterou C++ nabízí, byly tímto způsobem naprogramovány všechny kritické části a komponenty hry. Definovali jsme zde například všechny vlastnosti (omezující konstanty), jejichž konkrétní hodnoty jsme následně mohli měnit v Editoru skrze Blueprinty.

Nejvýznamější části:

1. Systém bloků a jejich správy
2. Komponenty *kyslíku*, *energie* a další
3. Bázové třídy pro UI elementy a jejich chování
4. Systém počasí
5. Systém ukládání a načítání hry
6. Systém inventáře

Blueprint

Hlavním cílem Blueprintů bylo prototypování funkcionality (většina z těchto prototypů byla následně implementována do C++) a rychlá implementace nekritických částí hry. Dále bylo Blueprintů využito jako nástroje, ve kterém byly definovány omezující konstanty, jako například minimální a maximální velikosti bloků, vlastnosti bloků, cena stavby apod. Díky tomu, že jsme tyto konstanty definovali na této úrovni, mohli jsme snadno balancovat jednotlivé parametry bez nutnosti zdlouhavého procesu opětovné komilace celého projektu.

Nejvýznamější části:

1. Definice bloků a jejich omezujících podmínek
2. AI systém pro řízení počasí
3. UI prvky a obrazovky
4. Systém shaderů pro škálovatelné bloky
5. Tutoriály
6. Systém ambientní hudby
7. Překlady pro anglickou jazykovou verzi

Obrázky ze hry

Lorem ipsum
obrazky

Závěr

Úspěšně se nám podařilo splnit všechny vytyčené cíle. Navrhli jsme a následně implementovali všechny námi požadované herní mechaniky, včetně doplňujících (lokalizace, hudba).

Tyto mechaniky jsme nechali posoudit veřejností v rámci připraveného dotazníku. Z jeho zhodnocení vyplývá, že myšlenka dynamicky škálovatelných bloků má svůj potenciál a případná dokončená hra by si našla své publikum. Zároveň jsme však díky dotazníku identifikovali problémy, které jsme mohli před dokončením naší práce vyřešit a tím jsme výrazně zlepšili celkovou kvalitu naší hry.