

Herní engine

Jiří Bittner

Přednáška začne ve 14:30!

Obsah přednášky

Co je herní engine?
GAE 1.3

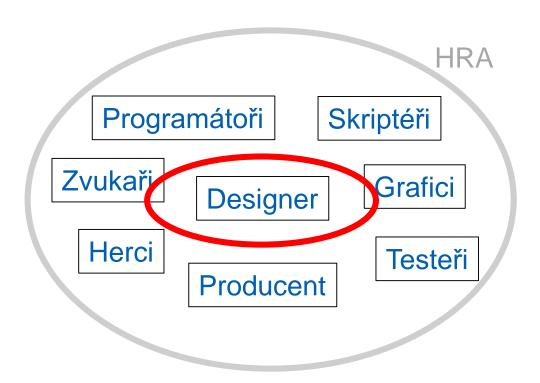
Herní enginy podle žánrů, přehled enginů GAE 1.4-1.5

Komponenty herního enginu

Herní smyčkaGPP 9, GEA 15.6

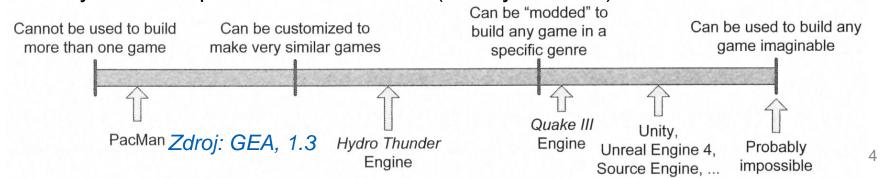
[GPP] R. Nystrom. Game Programming Patterns, 2014.

Vývoj hry - tým



Co je herní engine?

- Game engine = software pro tvorbu her
 - (soft) real-time dynamická interaktivní simulace
 - Typicky framework (předdefinovaná struktura volání fcí; prázdná hra)
- Historie
 - Do 1. pol. 90 let hry vždy vlastní "engine"
 - Možnost modifikací (mody ... Doom/Quake)
 - Oddělení enginu od herní mechaniky/gameplay/příběhu
 - Využitelnost pro tvorbu mnoha her (i různých žánrů)!



Herní enginy – přehled

- Quake engine (1996) Id Software
 - Quake, Quake II/III, Sin, ..., Medal of Honor: Pacific Assault
 - Zdrojové kódy od Quake and Quake II http://github.com/id-Software/Quake-2
- Unreal engine (1998) Epic Games
 - Unreal, Unreal Tournament, Gears of War, ...
 - I zdrojové kódy, UE4 skvělá platforma i pro VR
- Source engine (2004) Valve
 - Half-life 2, HL2: Episod One/Two, Counter Strike: Source, Team Fortress, Portal
- Frostbite (2006) DICE / EA
 - Battlefield Bad Company, Mass Effect, Need for Speed, Dragon Age, ...









Herní enginy - přehled

- Unity (2005) Unity Technologies
 - Wasteland 2, Battlestar Galactica Online, Temple Run, Assassin's Creed: Identity, Deus Ex: The Fall, ...
 - Parts of the engine open source (e.g. UI)
- CryENGINE (2006) Crytek
 - Far Cry, Crysis, Codename Kingdoms, Warface, Ryse: Son of Rome. ...
 - Open source
- Lumberyard (2015) Amazon (based on CryEngine)
 - Star Citizen, The Grand Tour Game, Open Source
- PhyreEngine (2008) Sony
 - Journey, Unravel, Demon's souls, flOw, ...
 - Open source









Herní enginy – přehled

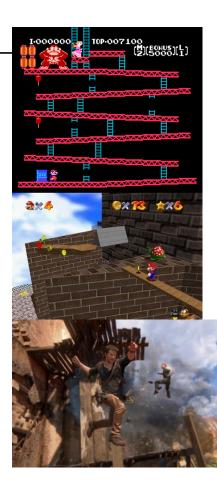
- In-house engines
 - Westwood Studios / EA Sage
 - Naughty Dog's engine
 - Bohemia Interactive Real Virtuality Engine
 - -
- Open Source
 - OGRE, Panda3D, Yake, Crystal Space, Torque, Irrlicht, ...
- Enginy pro neprogramátory
 - Multimedia Fusion2, Game Salad, Scratch, GameMaker, ...

- First-person-shooters (FPS)
 - Quake, Unreal Tournament, Half-life, Counter-Strike, Battlefield,...
 - Efektivní a kvalitní rendering rozsáhlých scén
 - Minimální zpoždění ovládání, kamery, zbraní
 - Kvalitní animace rukou a zbraní
 - Specializovaný pohyb prostorem (plavání přes překážky)
 - Kvalitní animace a umělá intelligence nepřátel (NPC)
 - Multiplayer v malém rozsahu (např. do 64 hráčů)
 - Deathmatch



Plošinovky (platformers) a 3rd person games

- 2D: Space panic, Donkey Kong, Pitfall!, Super Mario Bros, ...
- 3D: Super Mario 64, Rayman 2, Sonic the Hedgehog, Super Mario Galaxy, ...
- 3rd person: Dead Space 2, Gears of War 3, Uncharted, Resident Evil, ...
- Pohyblivé plošiny, žebříky, provazy, lešení, ...
- Prostředí ve formě skládačky, bloky
- Kamera z třetí osoby zaměřená na hráčovu postavu, ovládání rotace
- Komplexní kolizní systém pro kameru



Bojové hry

- Soul Calibur, Tekken 3, Fight Night Round 4, ...
- Soubojové animace
- Přesná detekce zásahů
- Komplexní kombinace vstupů: tlačítka + joystick
- Simulace obecenstva, jinak statické pozadí
- Grafika postav a věrné animace
- Shadery pro kůži, subsurface scattering, pot, ...
- Fyzikálně založené simulace oblečení a vlasů



Závodní hry

- Gran Turismo, Forza, WRC, ...
- Fyzikálně založená simulace vozidel
- Zobrazování pozadí (různé triky, vrstvy)
- Optimalizace trati (rozdělení na sektory)
- Umělá inteligence
- 3rd person kamera optimalizace záběru, detekce kolizí (tunel, ...)



Real-Time Strategie (RTS)

- Dune II, Warcraft, Command & Conquer, Age of Empires, Starcraft, ...
- Mřížka nebo obecný 3D svět
- Mnoho jednotek na obrazovce
- Zobrazování terénu
- Budování nových struktur
- Složitý interface klik + menu



- Massively Multiplayer Online Games (MMOG)
 - Guild Wars 2, EverQuest, World of Warcraft, Star Wars Galaxies, ...
 - Mnoho hráčů najednou (tisíce až stovky tisíc)
 - Různé žánry: MMORPG, MMORTS, MMOFPS
 - "Baterie" serverů, síťová komunikace, VOIP
 - Udržování konzistence herního světa



Další herní žánry

- Hry na hrdiny Role Playing Games (RPG)
 - Baldur's gate, The Witcher, Dark Souls, Dragon Age: Origins, Mass Effect, ...
- Sportovní hry
 - FIFA, NBA, NFL, PGA, ...
- Budovatelské / ekonomické strategie
 - Sim City, Caesar, Transport Tycoon
- Hráčem vytvářený obsah sandbox hry
 - Little Big Plannet, Minecraft, ...
- Hlavolamy puzzle games
 - Tetris, The Room, ...



Výběr enginu

- Velmi důležitá fáze projektu
- Kritéria
 - Podpora "game features"
 - Platforma (vývojová / herní)
 - Podpora různého HW
 - Učící křivka: možnosti enginu vs rychlost ovládnutí
 - Vývojářské rozhraní, programovací jazyk
 - Kvalita výstupu: rychlost zobrazování, animace, shadery, postprocess, ...
 - Kvalita dokumentace
 - Velikost vývojářské komunity
 - Integrace produktů a knihoven třetích stran
 - Dostupnost zdrojového kódu

Srovnání Unity a Unreal Engine [BP A. Šmíd 2017]

15

Obsah přednášky

Co je herní engine?

GAE 1.3

Herní enginy podle žánrů, přehled enginů

GAE 1.4-1.5

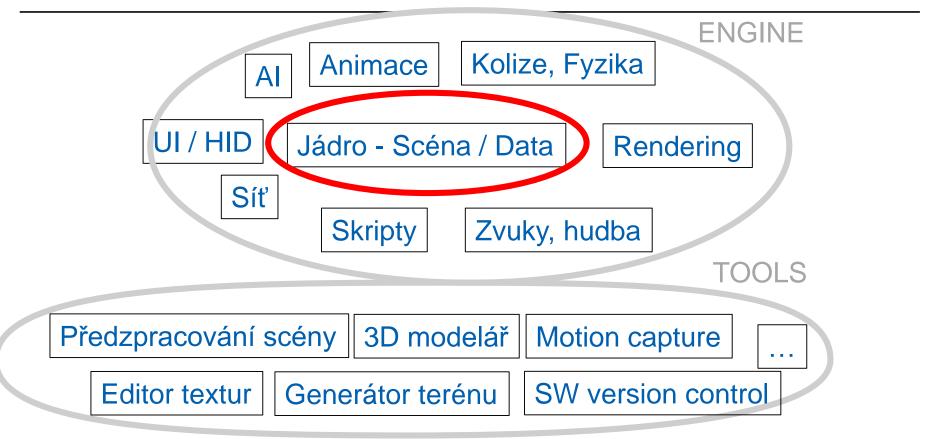
Komponenty herního enginu

Herní smyčka

GPP 9, GEA 15.6

[GPP] R. Nystrom. Game Programming Patterns, 2014.

Komponenty herního enginu



Herní engine - detailně

Game Specific!

- Vrstvy a komponenty
- GEA 1.6



Core Systems

Platform Independence
3rd party SDKs

Water Simulation Player-Follow Sight Traces & Terrain Rendering Collision Manifold Path Finding (A*) **Gameplay Foundations** Heads-Up Display Full-Motion Vide In-Game Cinematics High-Level Game Flow System/FSM (HUD) (EMM) (IGC) Wrappers / Attract Scripting System In-Game GUI In-Game Menus Static World Real-Time Agent Event/Messaging World Loading Elements **Object Model** Based Simulation Streaming Visual Effects Light Mapping & PRT Lighting, HDR Lighting Online Multiplayer Dynamic Shadows Subsurf, Scatte Object Attachment Animation State Match-Making & Particle & Decal Environment Game-Specific DSP/Effects Tree & Lavers Game Momt. Kinematics (IK) Post-Processino Systems Manning LERP and 3D Audio Model Scene Graph/ Culling Optimizations Additive Blendin Playback Animation Spatial Subdivision Game State Audio Playback Occlusion & PVS LOD System (BSP Tree &d-Tree ... Decompression Management Skeletal Mesh Rendering Low-Level Renderes Profiling & Debugging Collision &Physics Human Interfac Devices (HID) Materials & Static & Dynamic Recording & Forces & Ray/Shape Casting (Queries) Shaders Lighting Playback Constraints Primitive Viewports & Texture and Debug Drawing Memory & Game-Specific Rigid Bodies Phantoms Suhmission Virtual Screens Surface Mgmt (Lines etc.) Performance State Interface Physics/Collision Physical Device In-Game Menus Graphics Device Interface Collidables or Console Resources (Game Assets) 3D Model Texture Material Font Physics Game Collision etc. World/Map Resource Resource Resource Resource Resource Resource Parameters Resource Manager Core Systems Module Start-Un Strings and Debug Printing Localization Unit Testing Memory Allocation and Shut-Down Hashed String Ids Services and Logging Profiling / Stats Engine Config Random Number Curves & Object Handles Asynchronous Memory Card I/O Generator Surfaces Library Unique Ids (Older Consoles) Platform Independence Layer Atomic Data Collections and Physics/Coll. Threading Library Types Iterators Layer (UDP/TCP) Wrappers Wrapper 3rd Party SDKs DirectX, OpenGL, Havok, PhysX Granny, Hayok STL/STLPort Boost++ Euphoria Animation, etc. libgam, Edge, etc Hardware (PC, XBOX360, PS3, etc.)

GAME-SPECIFIC SUBSYSTEMS

Vahirles

Game Cameras

Scripted/Animated

Goals & Decision

Making

Actions

(Engine Interface)

Power-Hos

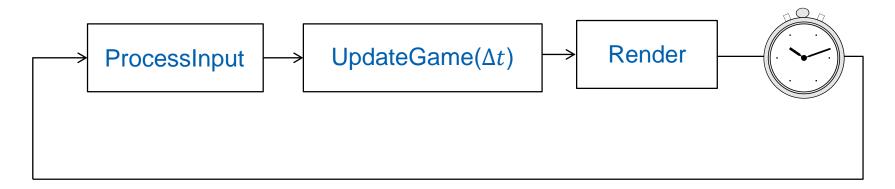
Camera-Relative

Controls (HID)

Player Mechanics
State Machine & Car



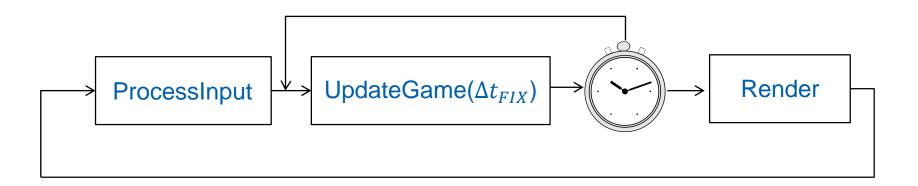
- Jak rychle smyčka poběží?
- Bez kontroly nad FPS a rychlostí běhu hry (herní čas) !



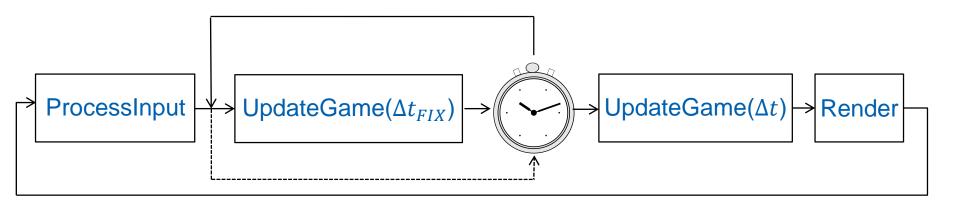
- UpdateGame(Δt) game tick
- Volitelná snímková synchronizace (např. 60FPS)
- Není čas jako čas!
 - Skutečný čas (real-time, wall-clock time)
 - Herní čas (logický čas ve hře mohu zrychlit, zpomalit, zastavit!)
 - CPU time (např. doba vykonávání funkce)



- Na různém HW velmi rozdílné Δt!
- Nedeterministické chování hry
 - Fyzika a kolize (a občas i herní logika a animace) požadují konstantní Δt



- Stabilní čas aktualizace Δt_{FIX}
- Aktualizujeme s Δt_{FIX} dokud simulovaný čas nedožene reálný čas
- Doba výpočtu UpdateGame(Δt_{FIX}) < Δt_{FIX} !



• Kombinace Δt_{FIX} a Δt

Herní smyčka – Pseudokód

```
double previous = GetCurrentTime();
double lag = 0.0;
while (true) {
 double current = GetCurrentTime();
 double elapsed = current – previous;
 previous = current;
 lag += elapsed;
 ProcessInput();
 while (lag >= \Delta t_{FIX}) {
  UpdateGameFixed(\Delta t_{FIX});
  lag = \Delta t_{FIX};
 UpdateGame (\Delta t);
 Render();
```

UpdateGame(Δt) - Aktualizace herních objektů

```
UpdateGame(\Delta t)
foreach o in objects
if (o.isActive)
o.Update(\Delta t)
```

Update je volána v každém snímku právě jednou

Aktualizace herních objektů

- Pozor na dlouhá těla funkcí!
 - Unity: Používat CoRoutines (yield WaitForSeconds(), yield null) ...

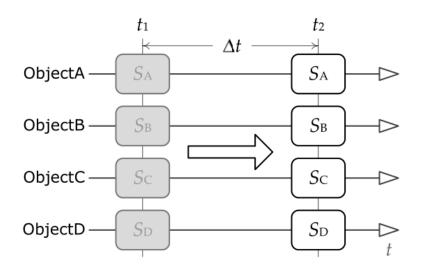
```
void Update() {
if (Input.GetKeyDown("f")) {
  StartCoroutine("Fade");
} }
```

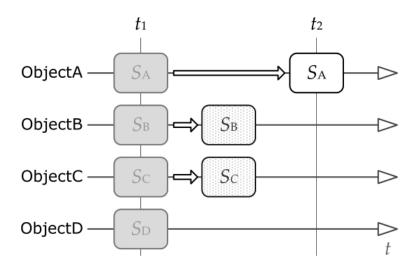
```
IEnumerator Fade() {
  for (float ft = 1f; ft >= 0; ft -= 0.1f) {
    Color c = renderer.material.color;
    c.a = ft;
    renderer.material.color = c;
    yield return new WaitForSeconds(.1f);
} }
```

- Mnoho objektů je neaktivních nebo statických
 - Udržovat explicitní seznam aktivních objektů!
 - Unity: GameObject.SetActive(bool)

Aktualizace herních objektů

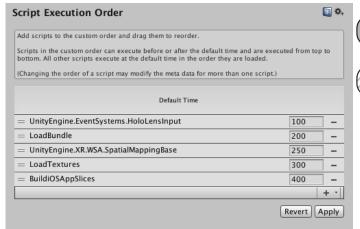
- Posloupnost aktualizací?
 - Důležitá pokud existují závislosti mezi objekty
 - Např. aktualizace parametrů kamery sledující několik NPC



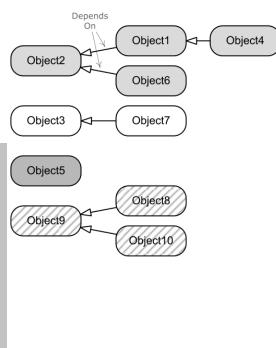


Aktualizace objektů - řešení závislostí

- Nalezení závislostí
 - Aktualizace nezávislých komponent v jedné dávce (i paralelní)



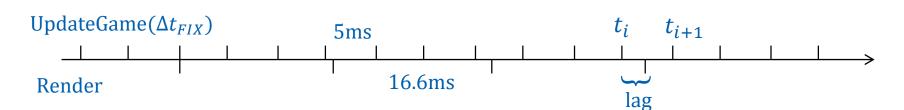
- Unity
 - Update + LateUpdate
 - ScriptExecutionOrder
 - vlastní UpdaterScript: explicitní volání metod objektů v daném pořadí



Aktualizace objektů - řešení závislostí

- Cache stavu objektů
 - Aktuální a minulý snímek
 - Data z minulého snímku jsou vždy dostupná!
- Cache se hodí i pro přesnou synchronizaci zobrazování (viz další slajd)
- Nevýhoda zvýšení paměťové náročnosti

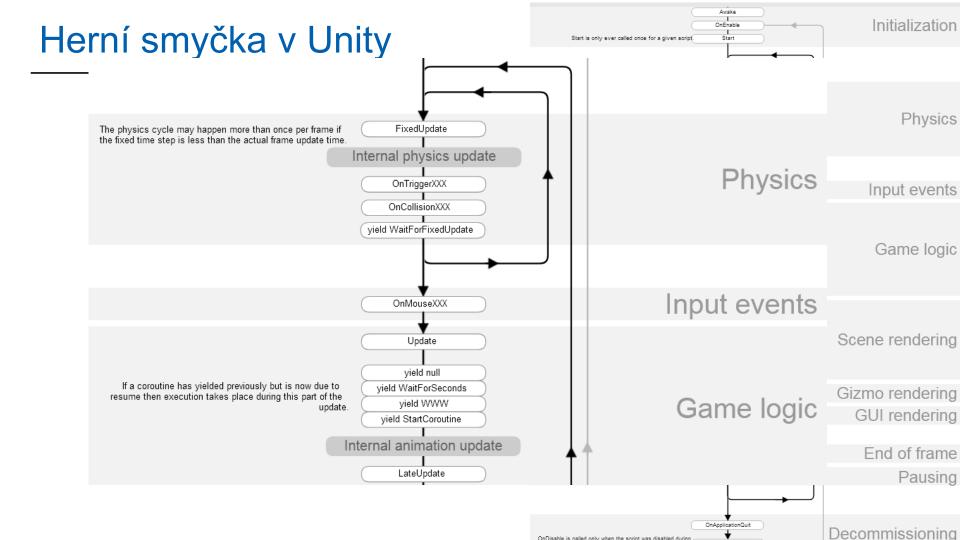
Herní smyčka – synchronizace zobrazování



- Zobrazovací čas není plně synchronizován se simulací
 - Pokud Δt_{FIX} dostatečně malé možno ignorovat
- Řešení 1
 - "Předběhnout" reálný čas: while (lag >= Δt_{FIX}) \rightarrow while (lag > 0)
 - Interpolace z posledních stavů: Render(LERP($t_i, t_{i+1}, -lag/\Delta t_{FIX}$)) (Unity: Rigidbody.interpolation = true)
- Řešení 2
 - Extrapolace stavu scény na základě znalosti rychlosti pohybu ve stavu t_i
 - Render(Extrapolate(t_i , lag))

Aktualizace komponent enginu

- Interní komponenty herního enginu se aktualizují jako celek
 - Fyzika / Kolize
 - Animace
 - Audio
 - Renderer
- Vazba na aktualizace objektů pomocí událostí
 - OnCollision, OnRender, OnPreRender, ...





Otázky?