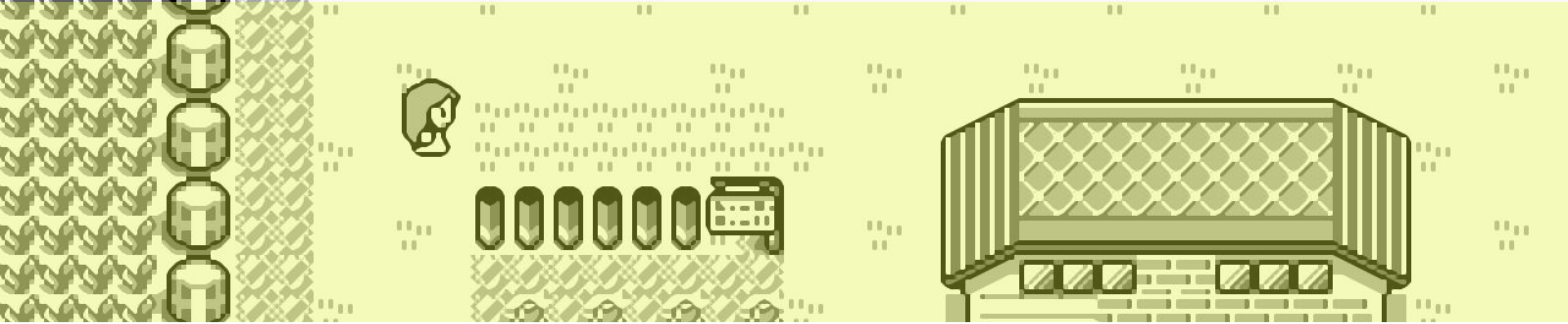
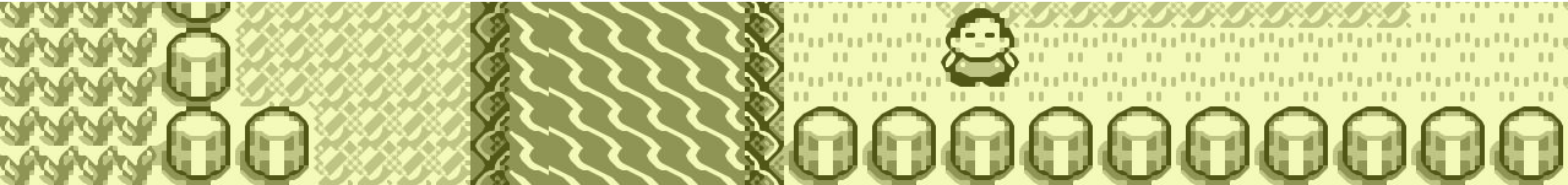


Game Tools

MARY BETH KERY - ADVANCED USER INTERFACES SPRING 2017



Part 1: Video game are complex software!!!

2 person team
3 years

300 person team
10 years

Final Fantasy 15

ART

GAME DESIGN

ENGINEERING

PRODUCTION/BUSINESS

TECHNICAL CHALLENGES OF VIDEO GAMES

1. **Video games are *real time* complex simulations, and must be efficient.**

TECHNICAL CHALLENGES OF VIDEO GAMES

1. **Video games are *real time* complex simulations, and must be efficient.**

1999 Roller Coaster Tycoon written by one guy in x86 assembly language

TECHNICAL CHALLENGES OF VIDEO GAMES

1. **Video games are *real time* complex simulations, and must be efficient.**

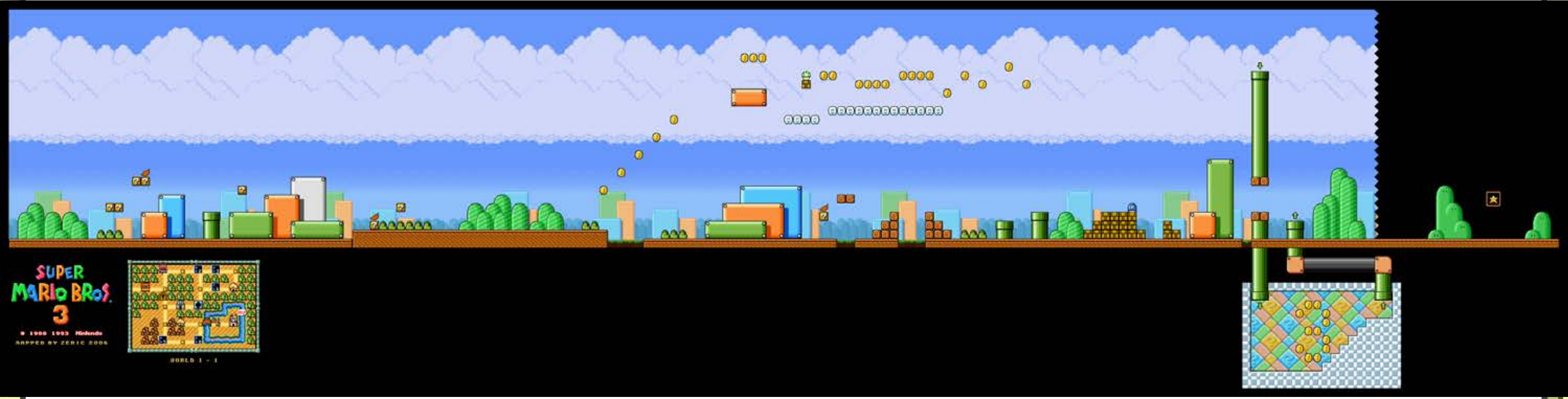
Today, more flexibility in language

Typically Object-Oriented

Use development tools like Visual Studio or Eclipse

TECHNICAL CHALLENGES OF VIDEO GAMES

2. People have high expectations for interactive worlds with lots of content



TECHNICAL CHALLENGES OF VIDEO GAMES

2. People have high expectations for interactive worlds with lots of content

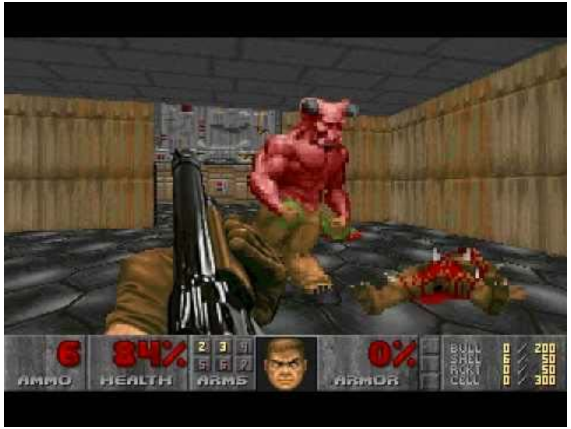
SKYRIM HAS THE BEST GLITCHES. I WILL ALWAYS LOVE YOU, SKY-MAMMOTH.
SKYRIMCONFESSIONS.COM

Lots of content on tight deadlines.

Glitches and crashes are BAD.

TECHNICAL CHALLENGES OF VIDEO GAMES

3. Real time 3D graphics simulations



Doom 1993

Levels, dungeons, and rooms were not only for game pacing, but to limit the number of objects to compute and render at a time.

TECHNICAL CHALLENGES OF VIDEO GAMES

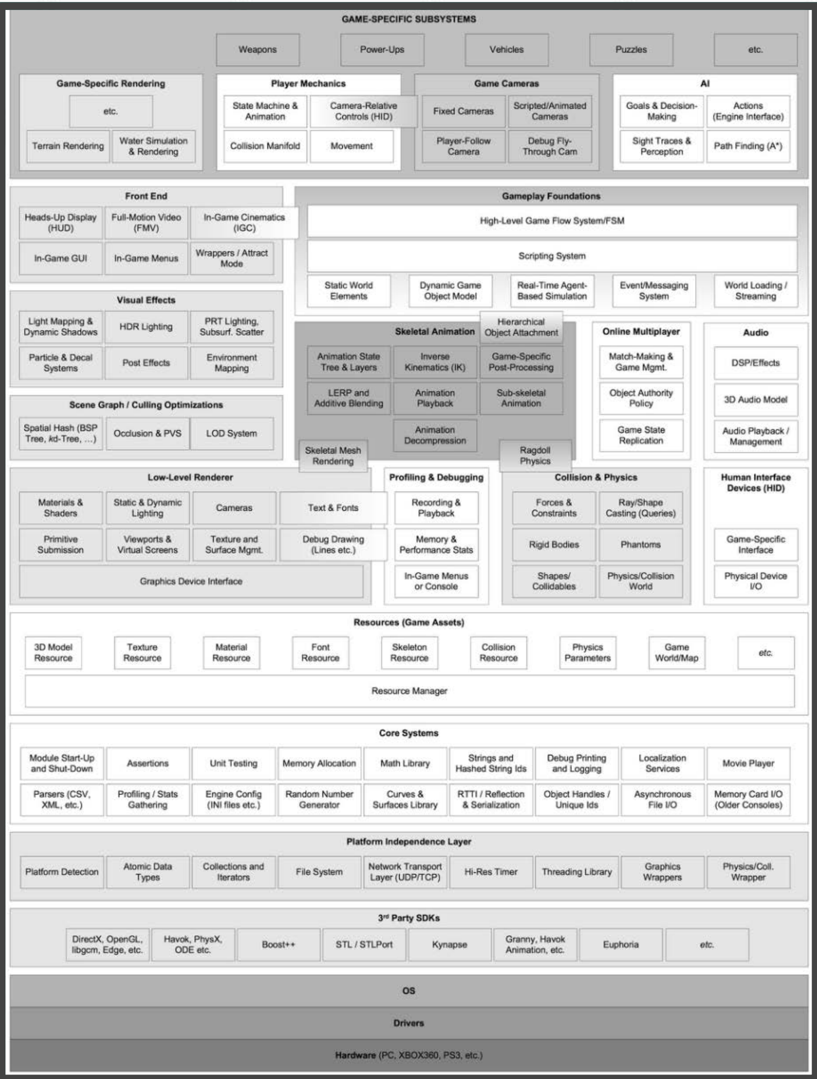
3. Real time 3D graphics simulations

2016 graphics

Pixar - Piper

SPEED
LIMIT
30
RHS-113

Final Fantasy 15



GAME-SPECIFIC SUBSYSTEMS

- Weapons
- Power-Ups
- Vehicles
- Puzzles
- etc.

Game-Specific Rendering

- etc.
- Terrain Rendering
- Water Simulation & Rendering

Player Mechanics

- State Machine & Animation
- Camera-Relative Controls (HID)
- Collision Manifold
- Movement

Game Cameras

- Fixed Cameras
- Scripted/Animated Cameras
- Player-Follow Camera
- Debug Fly-Through Cam

AI

- Goals & Decision-Making
- Actions (Engine Interface)
- Sight Traces & Perception
- Path Finding (A*)

Front End

- Heads-Up Display (HUD)
- Full-Motion Video (FMV)
- In-Game Cinematics (IGC)
- In-Game GUI
- In-Game Menus
- Wrappers / Attract Mode

Gameplay Foundations

- High-Level Game Flow System/FSM
- Scripting System
- Static World Elements
- Dynamic Game Object Model
- Real-Time Agent-Based Simulation
- Event/Messaging System
- World Loading / Streaming

Visual Effects

- Light Mapping & Dynamic Shadows
- HDR Lighting
- PRT Lighting, Subsurf. Scatter
- Particle & Decal Systems
- Post Effects
- Environment Mapping

Skeletal Animation

- Hierarchical Object Attachment
- Animation State Tree & Layers
- Inverse Kinematics (IK)
- Game-Specific Post-Processing
- LERP and Additive Blending
- Animation Playback
- Sub-skeletal Animation
- Animation Decompression
- Skeletal Mesh Rendering
- Ragdoll Physics

Online Multiplayer

- Match-Making & Game Mgmt.
- Object Authority Policy
- Game State Replication

Audio

- DSP/Effects
- 3D Audio Model
- Audio Playback / Management

Scene Graph / Culling Optimizations

- Spatial Hash (BSP Tree, *kd*-Tree, …)
- Occlusion & PVS
- LOD System

Low-Level Renderer

- Materials & Shaders
- Static & Dynamic Lighting
- Cameras
- Text & Fonts
- Primitive Submission
- Viewports & Virtual Screens
- Texture and Surface Mgmt.
- Debug Drawing (Lines etc.)
- Graphics Device Interface

Profiling & Debugging

- Recording & Playback
- Memory & Performance Stats
- In-Game Menus or Console

Collision & Physics

- Forces & Constraints
- Ray/Shape Casting (Queries)
- Rigid Bodies
- Phantoms
- Shapes/ Collidables
- Physics/Collision World

Human Interface Devices (HID)

- Game-Specific Interface
- Physical Device I/O

Resources (Game Assets)

- 3D Model Resource
- Texture Resource
- Material Resource
- Font Resource
- Skeleton Resource
- Collision Resource
- Physics Parameters
- Game World/Map
- etc.
- Resource Manager

Core Systems

- Module Start-Up and Shut-Down
- Assertions
- Unit Testing
- Memory Allocation
- Math Library
- Strings and Hashed String Ids
- Debug Printing and Logging
- Localization Services
- Movie Player
- Parsers (CSV, XML, etc.)
- Profiling / Stats Gathering
- Engine Config (INI files etc.)
- Random Number Generator
- Curves & Surfaces Library
- RTTI / Reflection & Serialization
- Object Handles / Unique Ids
- Asynchronous File I/O
- Memory Card I/O (Older Consoles)

Platform Independence Layer

- Platform Detection
- Atomic Data Types
- Collections and Iterators
- File System
- Network Transport Layer (UDP/TCP)
- Hi-Res Timer
- Threading Library
- Graphics Wrappers
- Physics/Coll. Wrapper

3rd Party SDKs

- DirectX, OpenGL, libgcm, Edge, etc.
- Havok, PhysX, ODE etc.
- Boost++
- STL / STLPort
- Kynapse
- Granny, Havok Animation, etc.
- Euphoria
- etc.

OS

Drivers

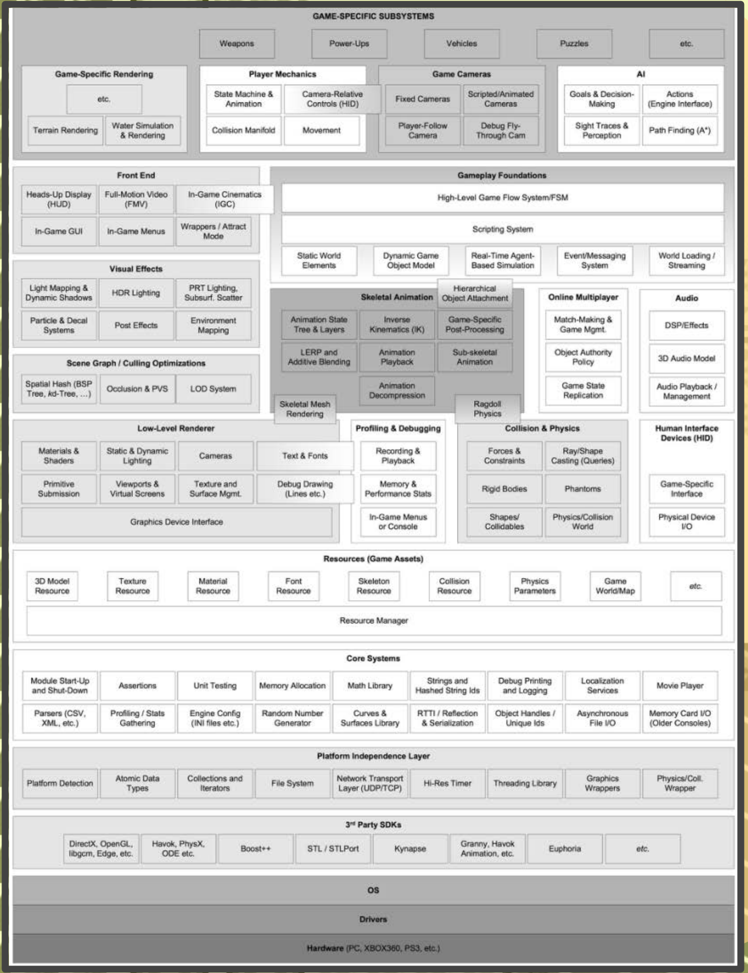
Hardware (PC, XBOX360, PS3, etc.)

Gregory, Jason. *Game engine architecture*. CRC Press, 2009.

Game Engines

Tools that fit the pieces together

Game Engine



GAME-SPECIFIC SUBSYSTEMS

Weapons
Power-Ups
Vehicles
Puzzles
etc.

Game-Specific Rendering
etc.
Terrain Rendering
Water Simulation & Rendering

Player Mechanics
State Machine & Animation
Camera-Relative Controls (HID)
Collision Manifold
Movement

Game Cameras
Fixed Cameras
Scripted/Animated Cameras
Player-Follow Camera
Debug Fly-Through Cam

AI
Goals & Decision-Making
Actions (Engine Interface)
Sight Traces & Perception
Path Finding (A*)

Front End
Heads-Up Display (HUD)
Full-Motion Video (FMV)
In-Game Cinematics (IGC)
In-Game GUI
In-Game Menus
Wrappers / Attract Mode

Gameplay Foundations
High-Level Game Flow System/FSM
Scripting System
Static World Elements
Dynamic Game Object Model
Real-Time Agent-Based Simulation
Event/Messaging System
World Loading / Streaming

Visual Effects
Light Mapping & Dynamic Shadows
HDR Lighting
PRT Lighting, Subsurf. Scatter
Particle & Decal Systems
Post Effects
Environment Mapping

Skeletal Animation
Hierarchical Object Attachment
Animation State Tree & Layers
Inverse Kinematics (IK)
Game-Specific Post-Processing
LERP and Additive Blending
Animation Playback
Sub-skeletal Animation
Animation Decompression
Skeletal Mesh Rendering
Ragdoll Physics

Online Multiplayer
Match-Making & Game Mgmt.
Object Authority Policy
Game State Replication

Audio
DSP/Effects
3D Audio Model
Audio Playback / Management

Scene Graph / Culling Optimizations
Spatial Hash (BSP Tree, *kd*-Tree, …)
Occlusion & PVS
LOD System

Low-Level Renderer
Materials & Shaders
Static & Dynamic Lighting
Cameras
Text & Fonts
Primitive Submission
Viewports & Virtual Screens
Texture and Surface Mgmt.
Debug Drawing (Lines etc.)
Graphics Device Interface

Profiling & Debugging
Recording & Playback
Memory & Performance Stats
In-Game Menus or Console

Collision & Physics
Forces & Constraints
Ray/Shape Casting (Queries)
Rigid Bodies
Phantoms
Shapes/ Collidables
Physics/Collision World

Human Interface Devices (HID)
Game-Specific Interface
Physical Device I/O

Resources (Game Assets)
3D Model Resource
Texture Resource
Material Resource
Font Resource
Skeleton Resource
Collision Resource
Physics Parameters
Game World/Map
etc.
Resource Manager

Core Systems
Module Start-Up and Shut-Down
Assertions
Unit Testing
Memory Allocation
Math Library
Strings and Hashed String Ids
Debug Printing and Logging
Localization Services
Movie Player
Parsers (CSV, XML, etc.)
Profiling / Stats Gathering
Engine Config (INI files etc.)
Random Number Generator
Curves & Surfaces Library
RTTI / Reflection & Serialization
Object Handles / Unique Ids
Asynchronous File I/O
Memory Card I/O (Older Consoles)

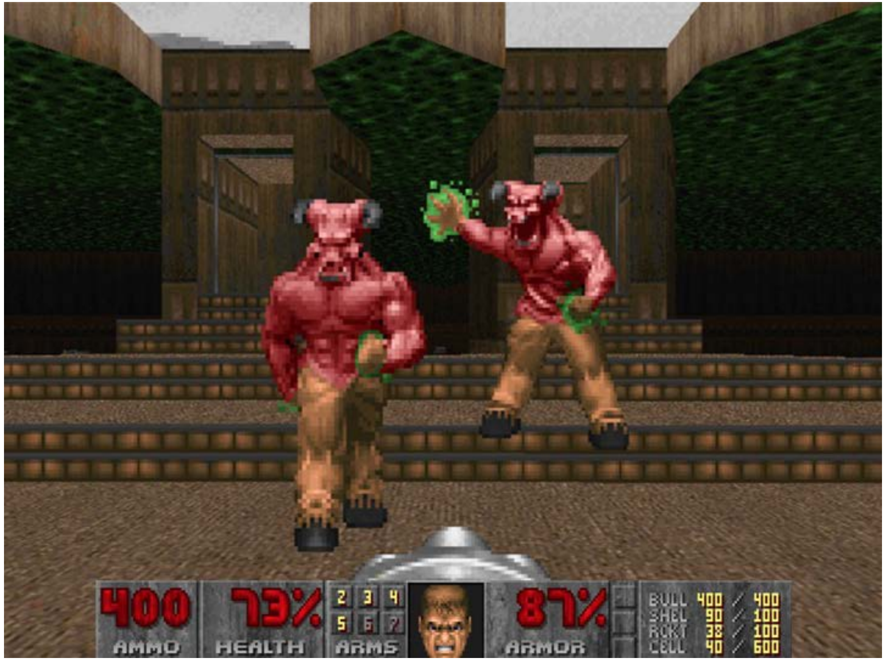
Platform Independence Layer
Platform Detection
Atomic Data Types
Collections and Iterators
File System
Network Transport Layer (UDP/TCP)
Hi-Res Timer
Threading Library
Graphics Wrappers
Physics/Coll. Wrapper

3rd Party SDKs
DirectX, OpenGL, libgcm, Edge, etc.
Havok, PhysX, ODE etc.
Boost++
STL / STLPort
Kynapse
Granny, Havok Animation, etc.
Euphoria
etc.

OS
Drivers
Hardware (PC, XBOX360, PS3, etc.)

GAME ENGINES: HISTORY

1990s First-person shooters: **Doom by id Software**



GAME ENGINES: HISTORY

Architecture separates core software from game-specific assets

ASSETS

- Art assets
- Game map/environments
- Rules of play

“ENGINE” SOFTWARE

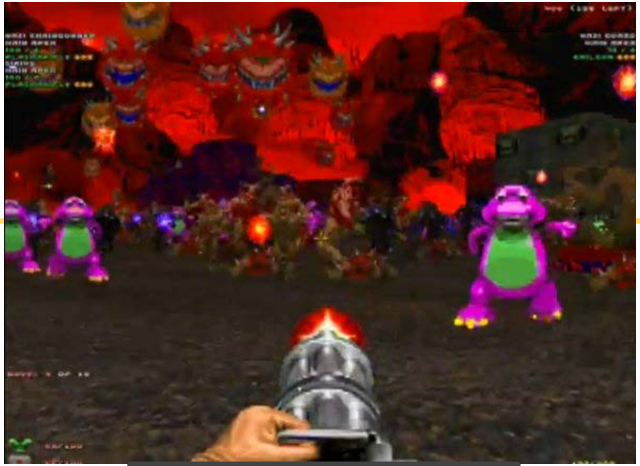
- 3D graphics rendering
- Collision detection
- Audio system

GAME ENGINES: HISTORY

1990’s Separation of game engine allowed “mods” by replacing assets

ASSETS

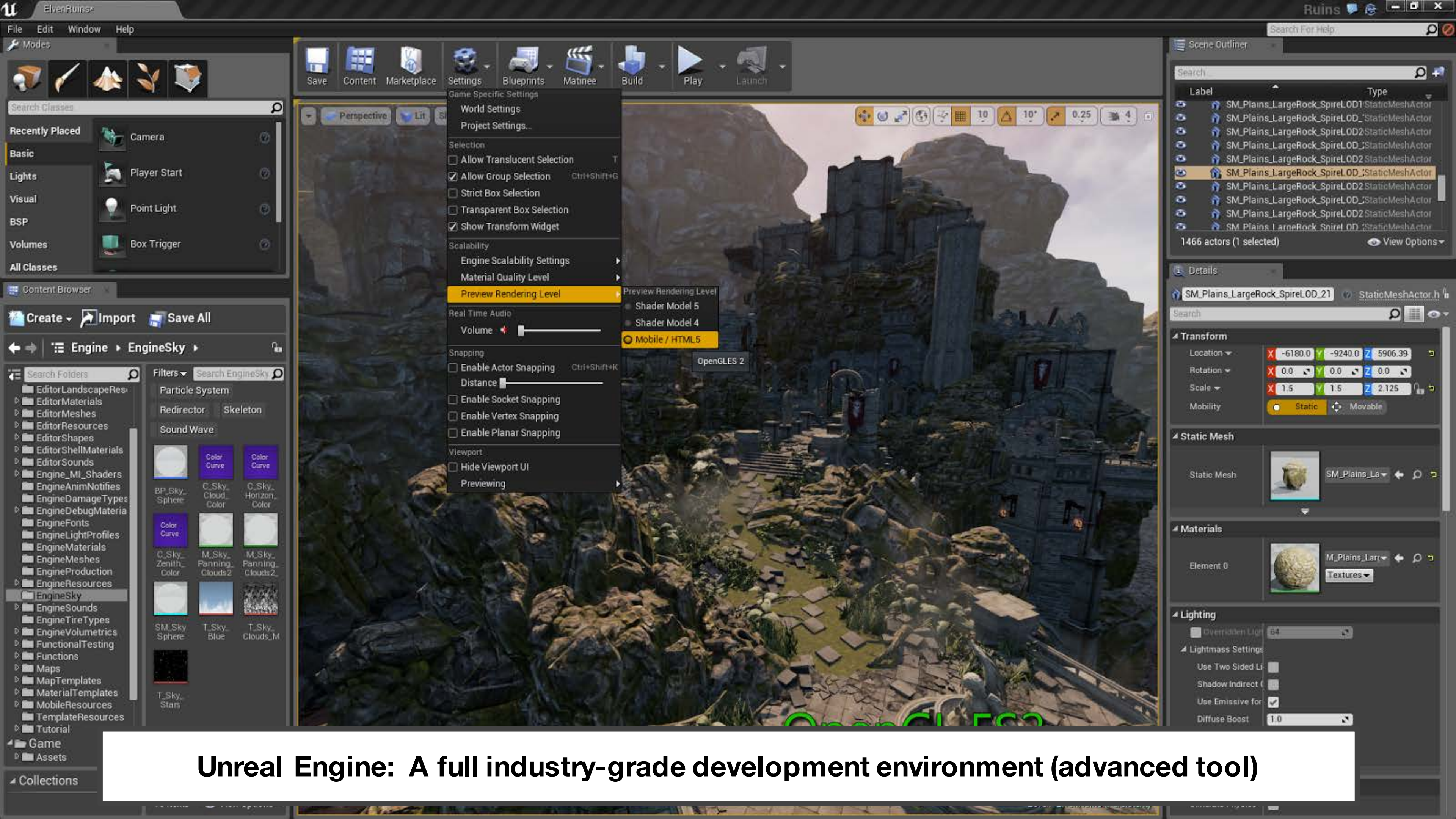
- Art assets
- Game map/environments
- Rules of play



Not okay mod.

“ENGINE” SOFTWARE

- 3D graphics rendering
- Collision detection
- Audio system



Game Specific Settings

World Settings

Project Settings...

Selection

☐ Allow Translucent Selection T

☑ Allow Group Selection Ctrl+Shift+G

☐ Strict Box Selection

☐ Transparent Box Selection

☑ Show Transform Widget

Scalability

Engine Scalability Settings

Material Quality Level

Preview Rendering Level

Real Time Audio

Volume

Snapping

☐ Enable Actor Snapping Ctrl+Shift+K

Distance

☐ Enable Socket Snapping

☐ Enable Vertex Snapping

☐ Enable Planar Snapping

Viewport

☐ Hide Viewport UI

Previewing

Preview Rendering Level

○ Shader Model 5

○ Shader Model 4

● Mobile / HTML5

Label	Type
SM_Plains_LargeRock_SpireLOD1	StaticMeshActor
SM_Plains_LargeRock_SpireLOD_	StaticMeshActor
SM_Plains_LargeRock_SpireLOD2	StaticMeshActor
SM_Plains_LargeRock_SpireLOD_	StaticMeshActor
SM_Plains_LargeRock_SpireLOD2	StaticMeshActor
SM_Plains_LargeRock_SpireLOD_	StaticMeshActor
SM_Plains_LargeRock_SpireLOD2	StaticMeshActor
SM_Plains_LargeRock_SpireLOD_	StaticMeshActor
SM_Plains_LargeRock_SpireLOD2	StaticMeshActor
SM_Plains_LargeRock_SpireLOD_	StaticMeshActor

1466 actors (1 selected)

SM_Plains_LargeRock_SpireLOD_21 StaticMeshActor.h

Location X -6180.0 Y -9240.0 Z 5906.39

Rotation X 0.0 Y 0.0 Z 0.0

Scale X 1.5 Y 1.5 Z 2.125

Mobility ● Static ○ Movable

Static Mesh SM_Plains_La

Element 0 M_Plains_Larg

☐ Overridden Light 64

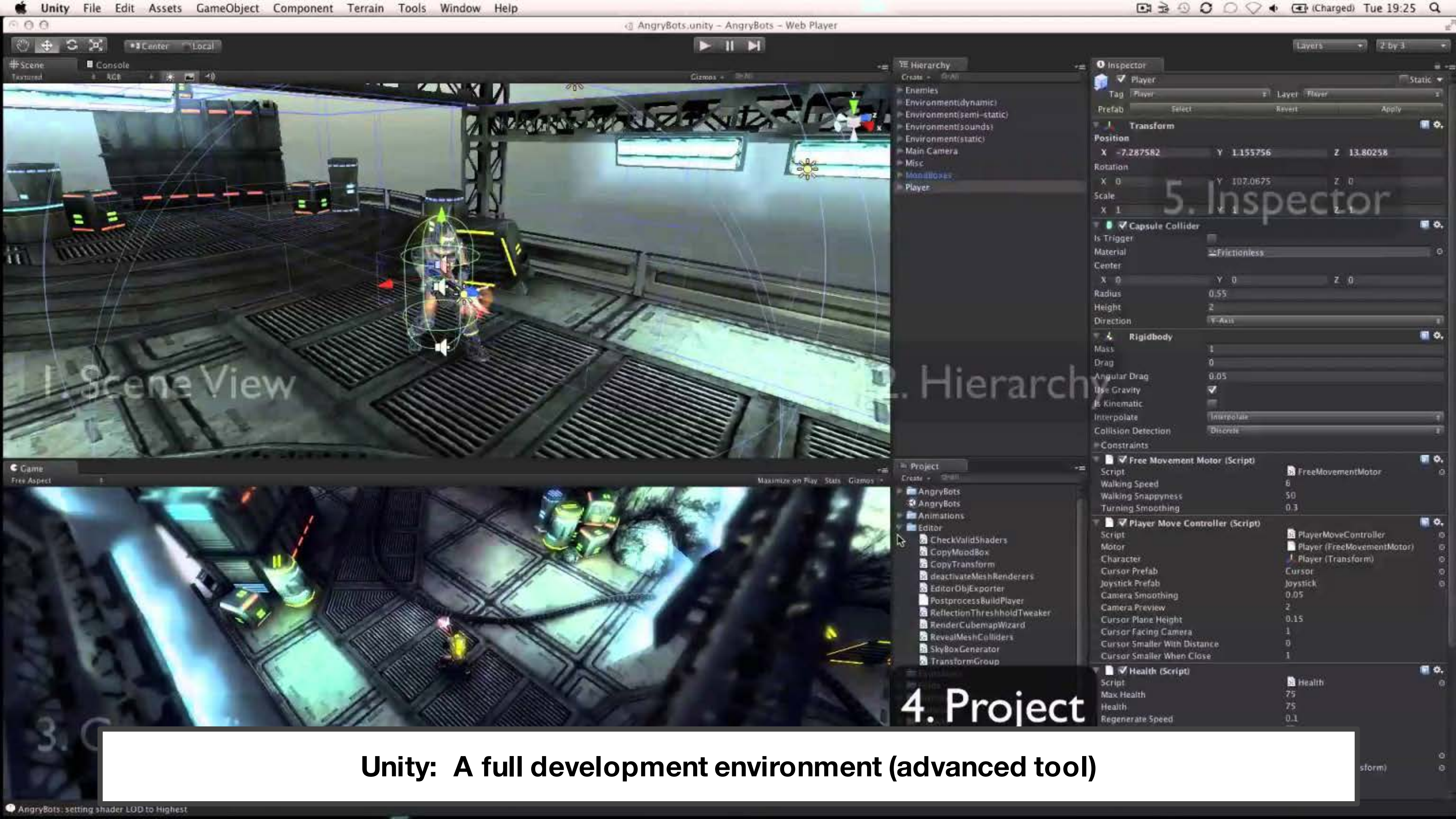
Use Two Sided Li ☐

Shadow Indirect ☐

Use Emissive for ☑

Diffuse Boost 1.0

Unreal Engine: A full industry-grade development environment (advanced tool)



Player
Static
Tag Player Layer Player
Prefab Select Revert Apply

Transform
Position X -7.287582 Y 1.155756 Z 13.80258
Rotation X 0 Y 107.0675 Z 0
Scale X 1 Y 1 Z 1

Capsule Collider
Is Trigger
Material Frictionless
Center X 0 Y 0 Z 0
Radius 0.55
Height 2
Direction Y-Axis

Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Use Gravity ✓
Is Kinematic
Interpolate Interpolate
Collision Detection Discrete
Constraints

Free Movement Motor (Script)
Script FreeMovementMotor
Walking Speed 6
Walking Snappyness 50
Turning Smoothing 0.3

Player Move Controller (Script)
Script PlayerMoveController
Motor Player (FreeMovementMotor)
Character Player (Transform)
Cursor Prefab Cursor
Joystick Prefab Joystick
Camera Smoothing 0.05
Camera Preview 2
Cursor Plane Height 0.15
Cursor Facing Camera 1
Cursor Smaller With Distance 0
Cursor Smaller When Close 1

Health (Script)
Script Health
Max Health 75
Health 75
Regenerate Speed 0.1

Unity: A full development environment (advanced tool)

A game engine has a data driven architecture that can be used to make many games

That dragon cancer

unity

Gardenarium

Clockwork

Art assets & animation

Graphics

Physics engines

Game loop

Art assets & animation

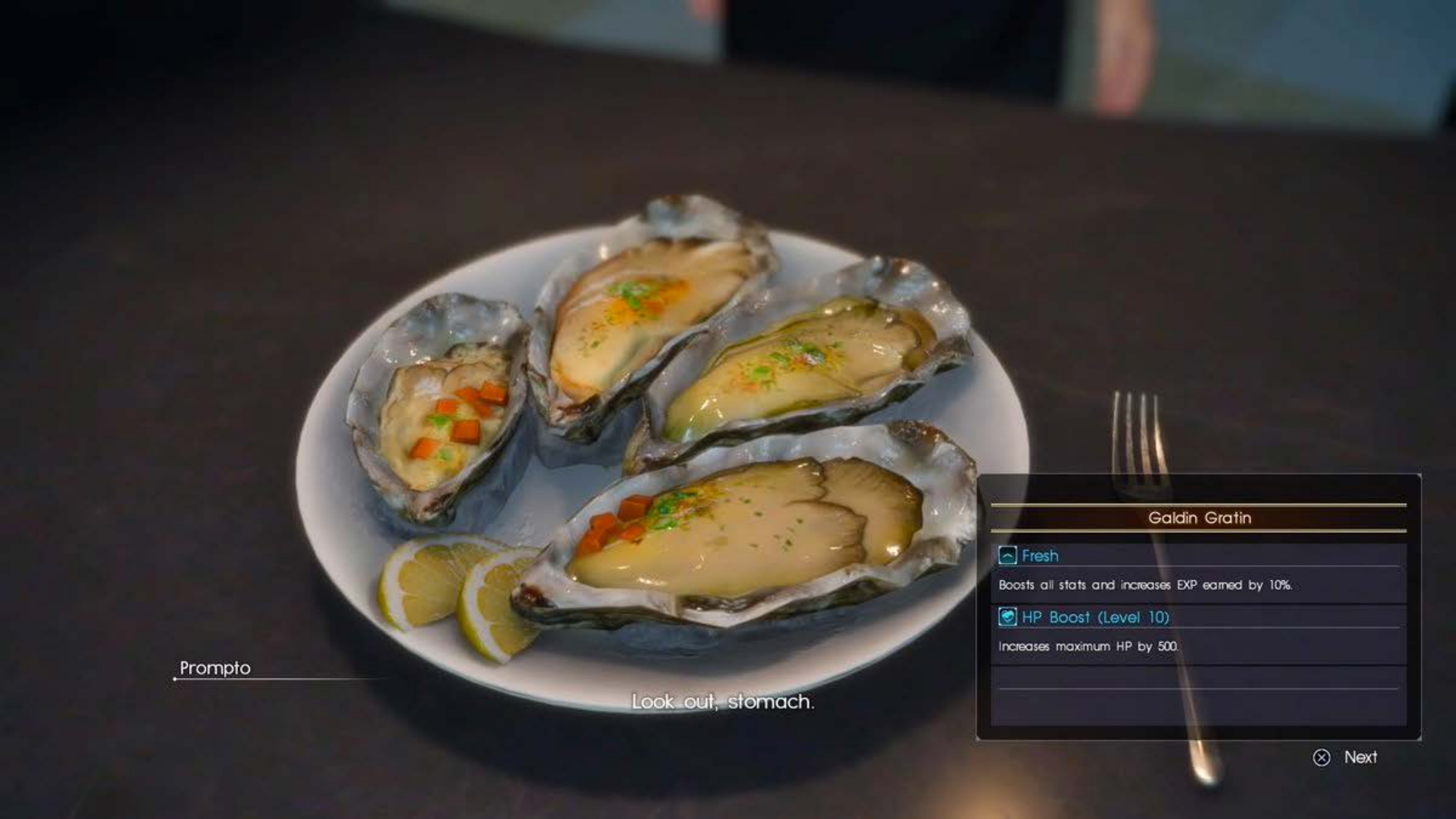
Graphics

Physics engines

Game loop

Art to game

Workflow of artists with tools and the game engine



Prompto

Look out, stomach.

Galdin Gratin

Fresh

Boosts all stats and increases EXP earned by 10%.

HP Boost (Level 10)

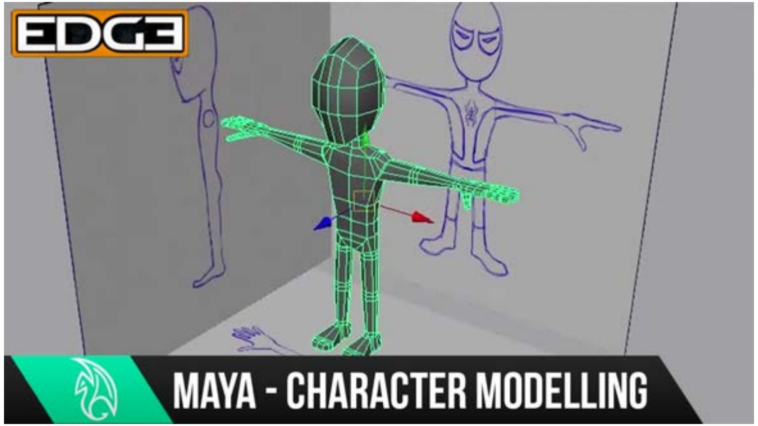
Increases maximum HP by 500.

Photo or drawing

The Final Fantasy 15 team cooked food and then photographed it as reference material for 3D modelers and shaders.

3D Scanning or image tracing

The Final Fantasy 15 team scanned their food and photographed it

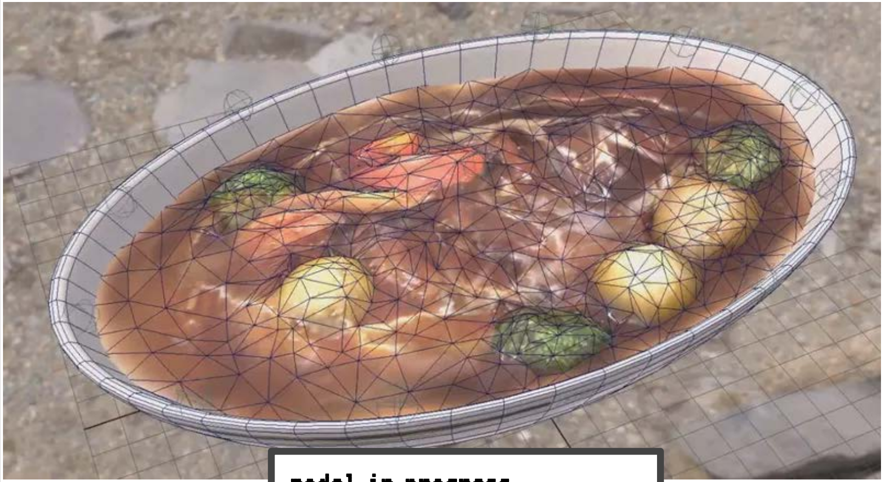


Modelers use reference drawings from different angles

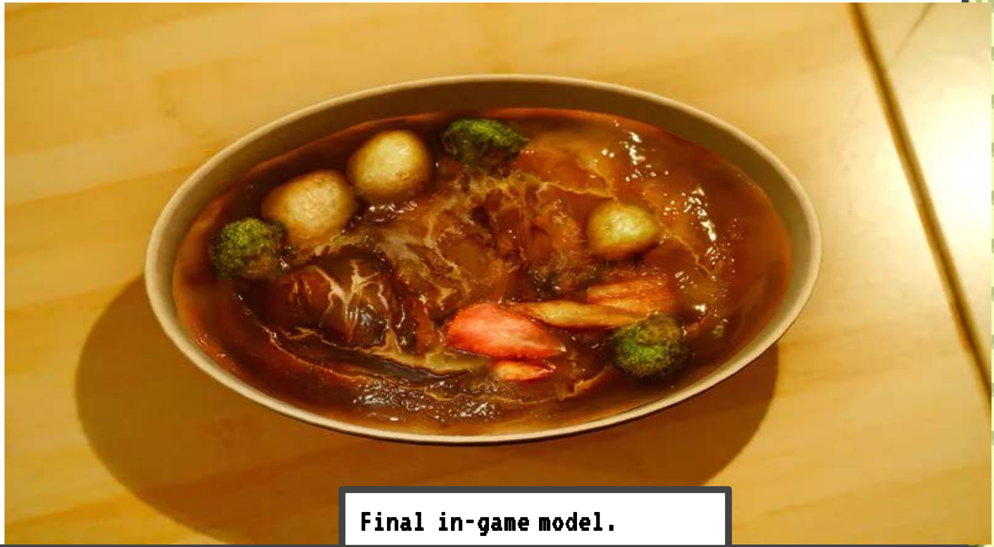
Modeling Software

AUTODESK® MAYA® 2015

blender™



model in progress

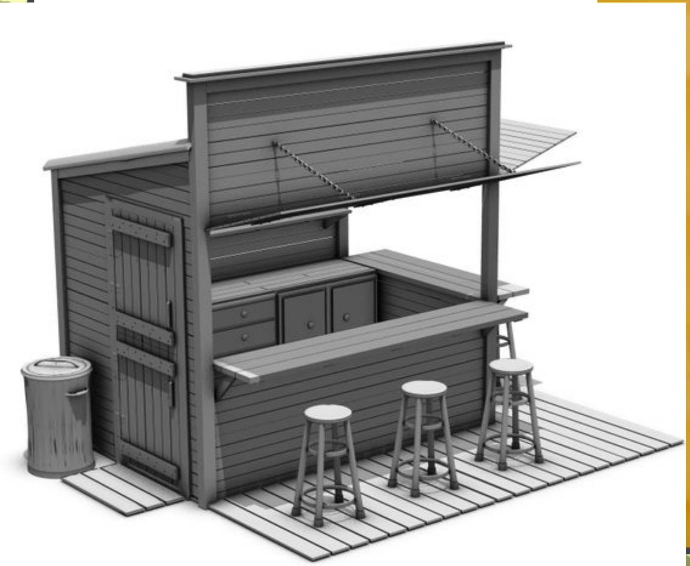


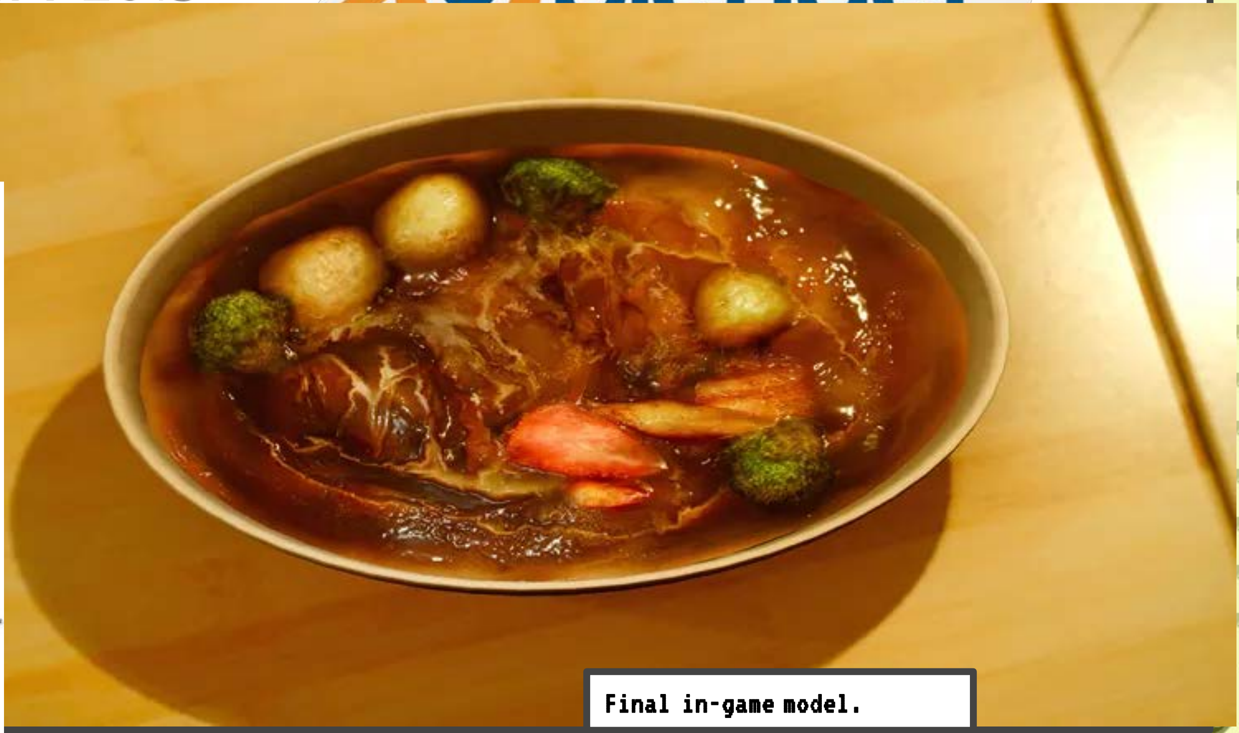
Final in-game model.

Textures and Shading

AUTODESK® MAYA® 2015

blender™

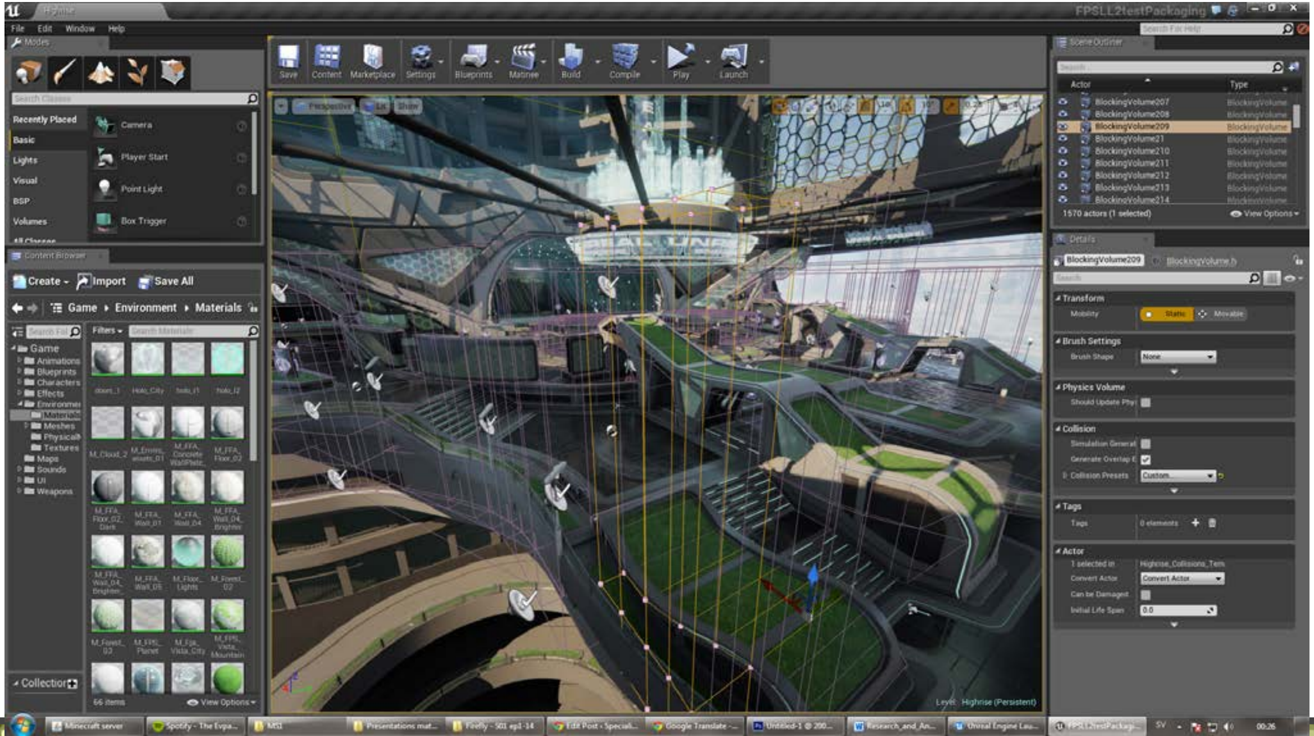




Final in-game model.

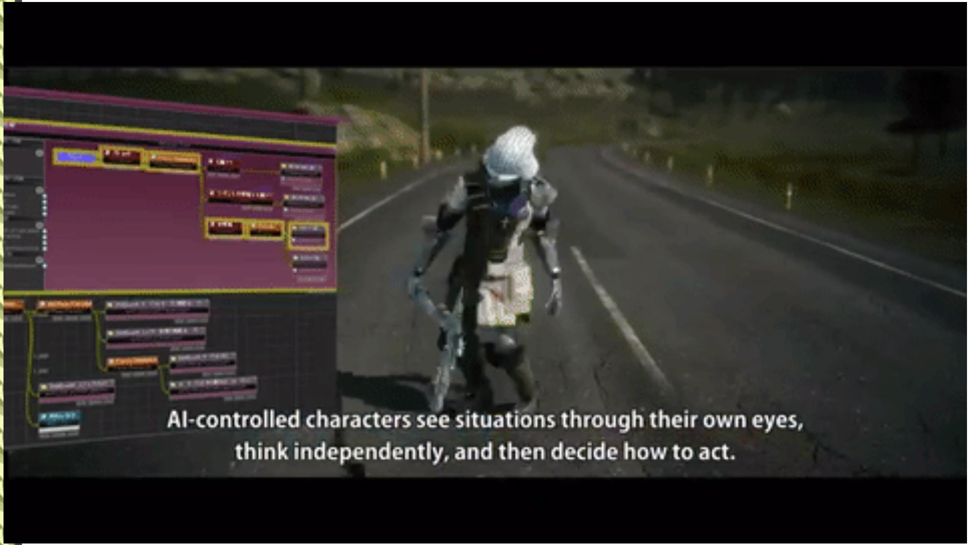
Back to the game

Unreal Engine place objects in scene with map editor

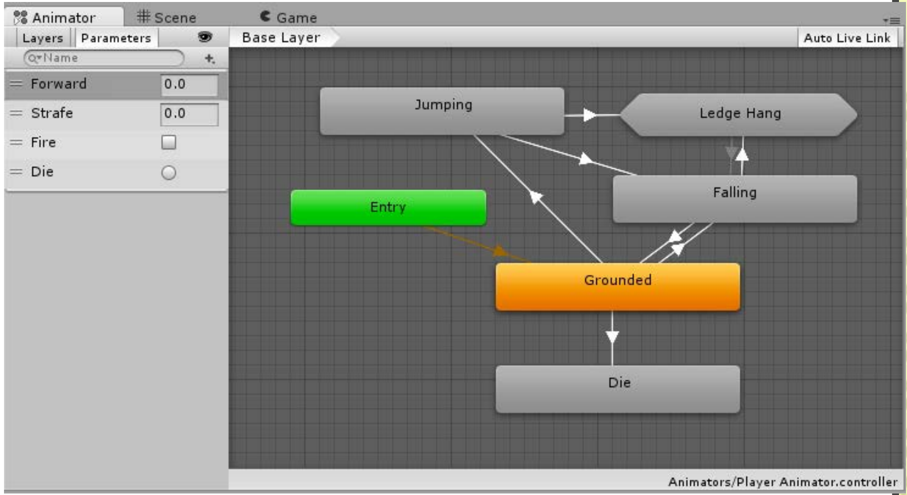


In the game engine

Visual programming languages allow animations, materials, and shaders to be written by artists



AI-controlled characters see situations through their own eyes, think independently, and then decide how to act.



Forward 0.0

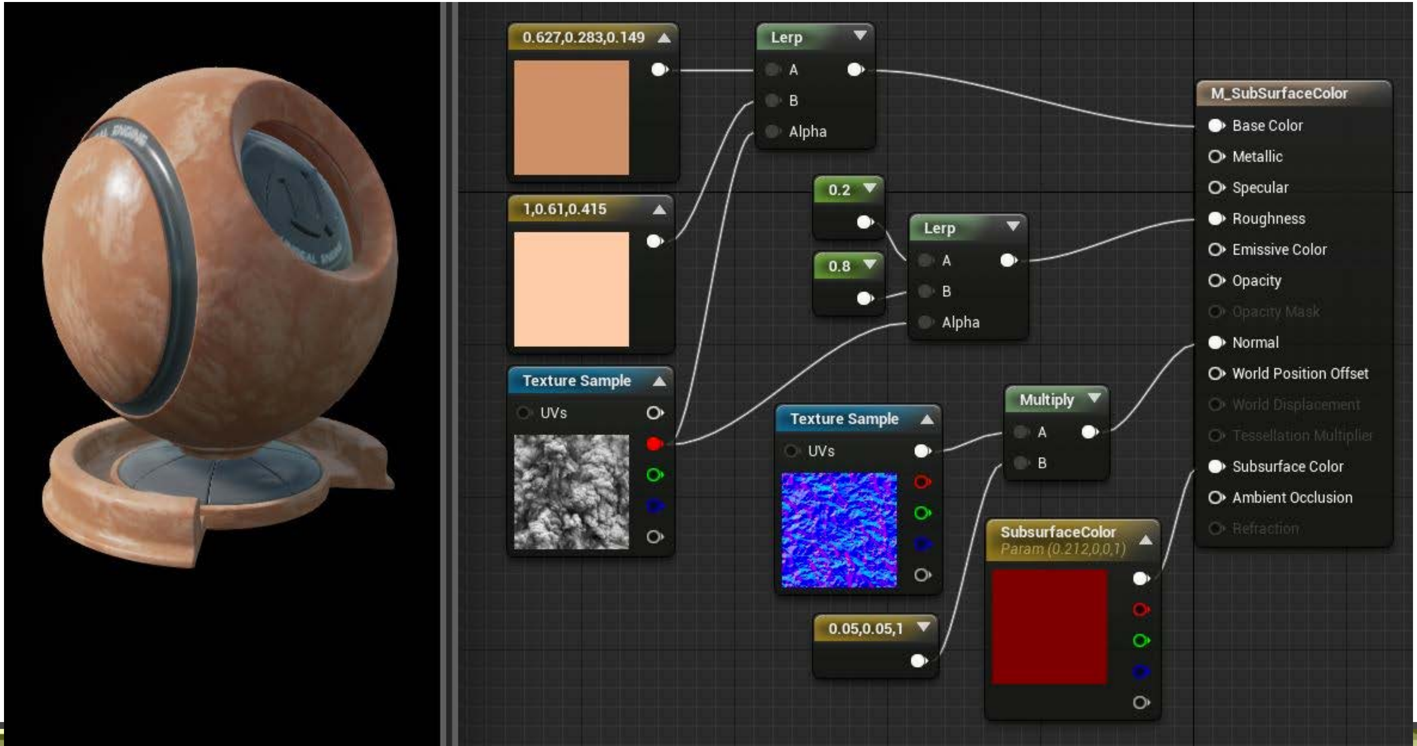
Strafe 0.0

Fire ☐

Die ○

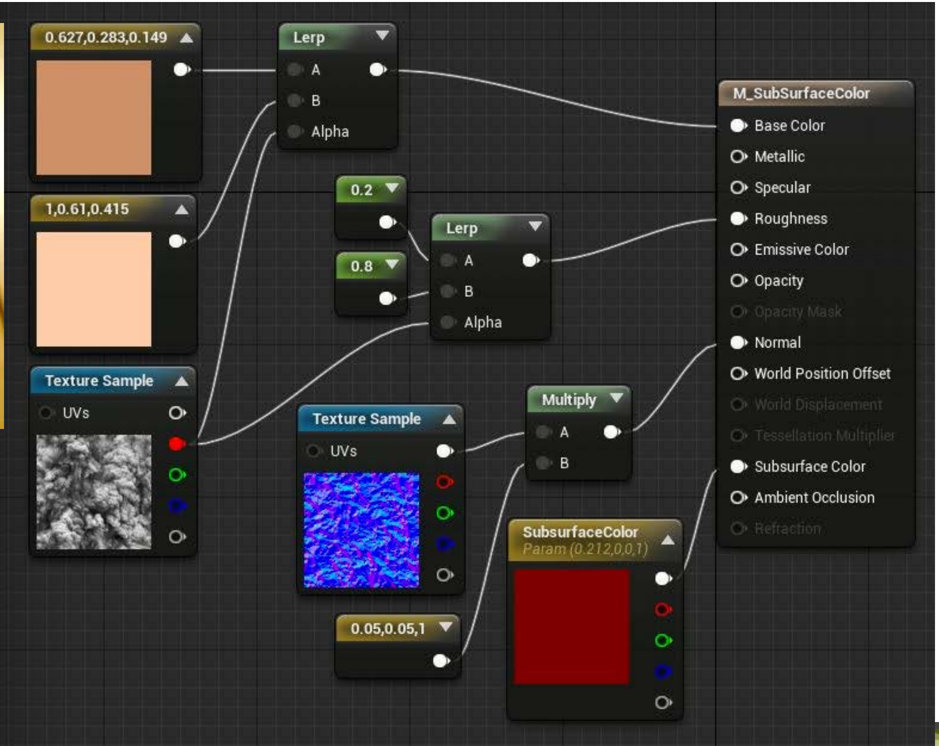
In the game engine

Visual programming languages allow animations, materials, and shaders to be written by artists



In the game engine

Visual programming languages allow animations, materials, and shaders to be written by artists



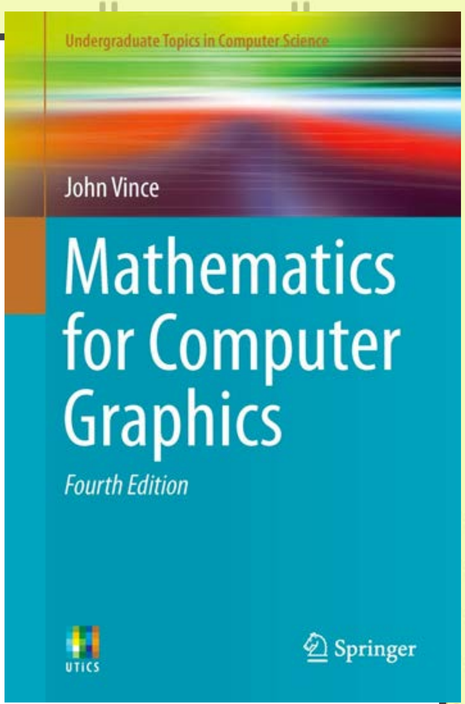
Art assets & animation

Graphics

Physics engines

Game loop

Shaders = VERY TECHNICAL



Undergraduate Topics in Computer Science

John Vince

Mathematics for Computer Graphics

Fourth Edition

UTiCS

Springer

COMPUTER GRAPHICS! 🎉

Technical Graphics Tools

OpenGL®

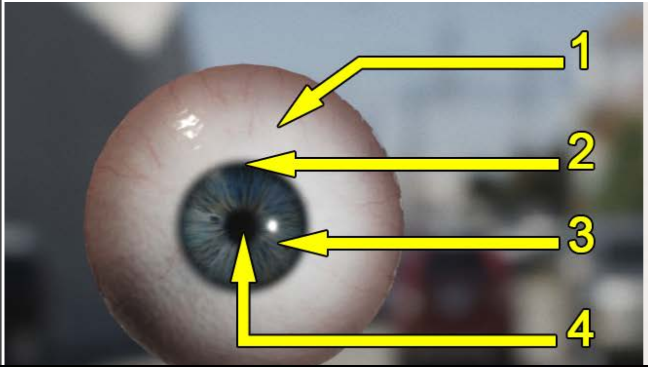
Open GL has bindings in lots of different languages

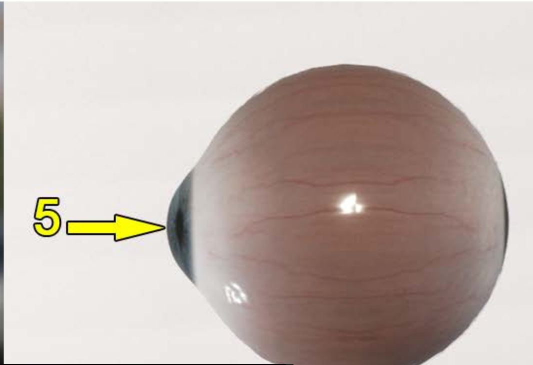
Powerful, but not easy to learn.

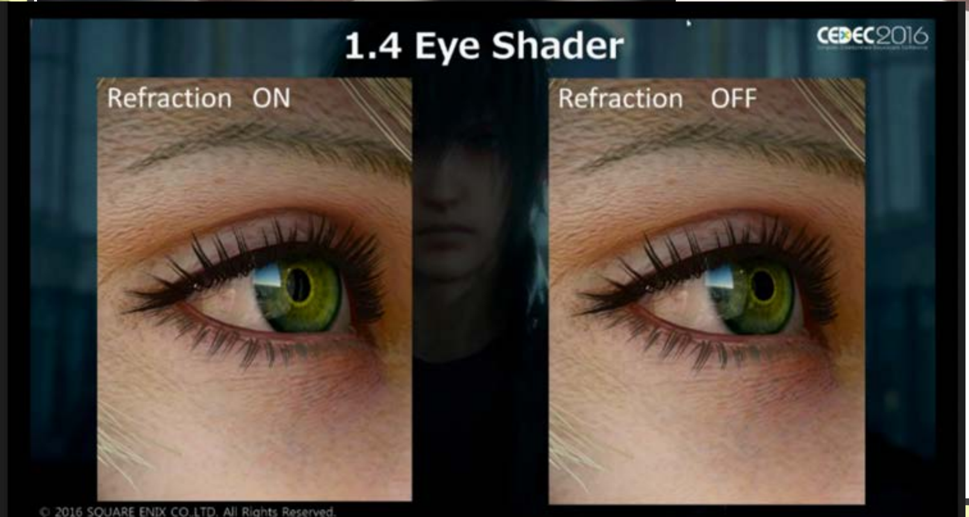
three.js

Some language bindings are more learner-friendly than others

Technical Graphics - eyes





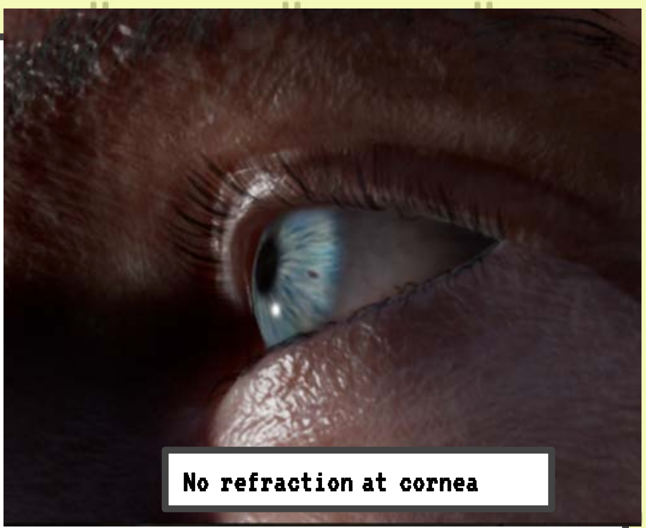


1.4 Eye Shader

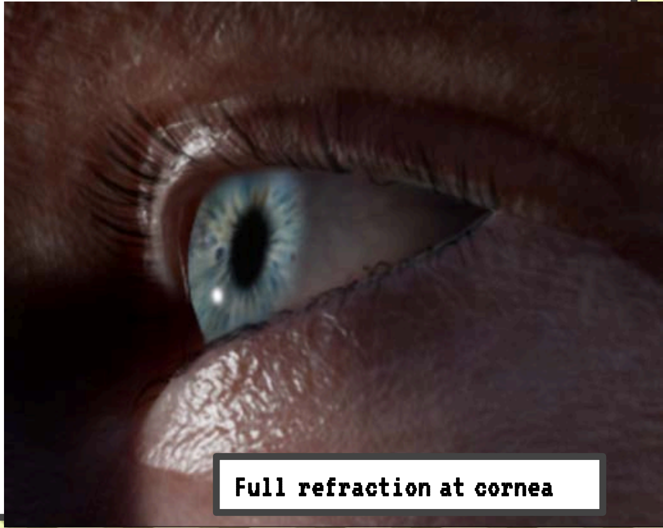
CEDEC2016

Refraction ON

Refraction OFF

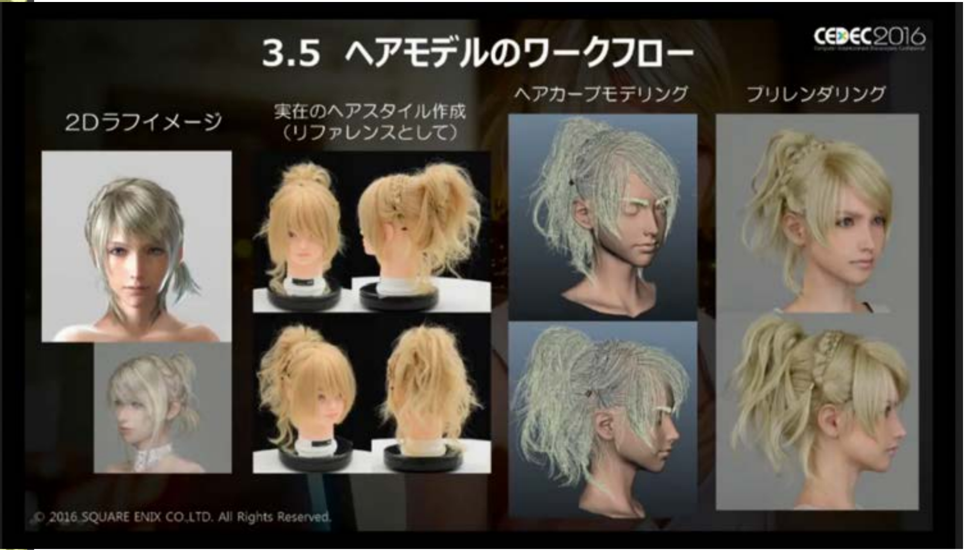


No refraction at cornea



Full refraction at cornea

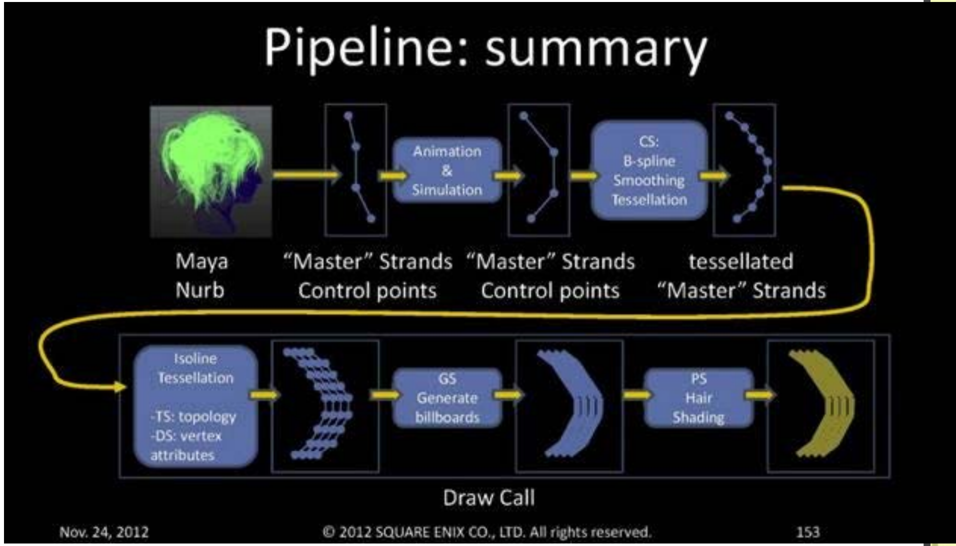
Technical Graphics - hair



3.5 ヘアモデルのワークフロー

CEDEC2016

© 2016 SQUARE ENIX CO.,LTD. All Rights Reserved.



Pipeline: summary

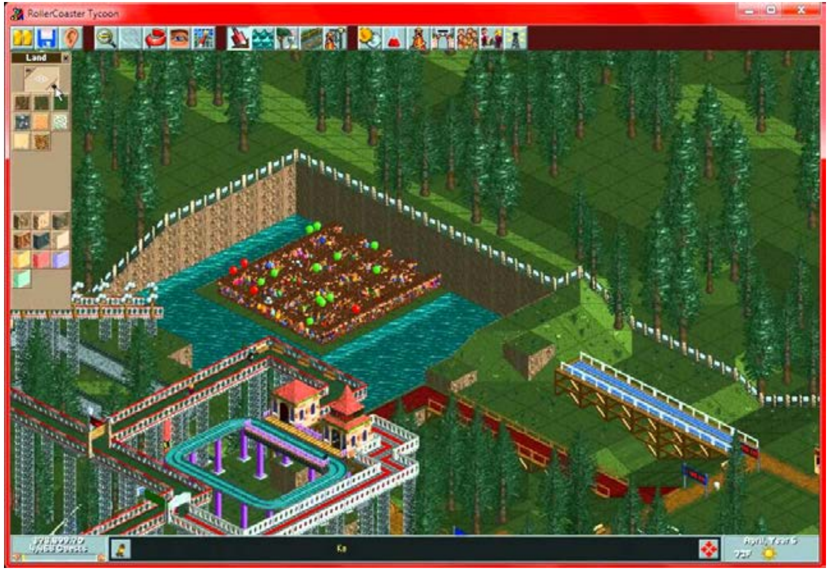
Nov. 24, 2012

© 2012 SQUARE ENIX CO., LTD. All rights reserved.

153

Process of modeling and rendering character Lunafreya's hair from Final Fantasy 15x

Graphics - Updating the Screen



Must be efficient!

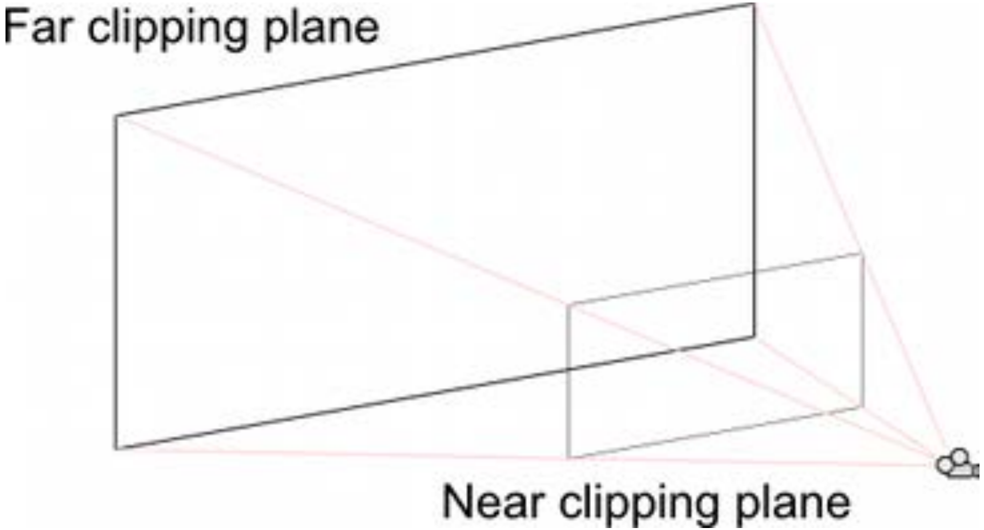
The screen must be updated every frame, at 30fps to 60fps. Rendering and shaders are computationally expensive!

Graphics - Updating the Screen

Occlusion culling problem: don’t render hidden objects

Frustum culling: test if an object intersects with the frustum.

Portals: designers *manually* place simple primitives around chunks of the game world. The portals are invisible but cheap to test intersection on.

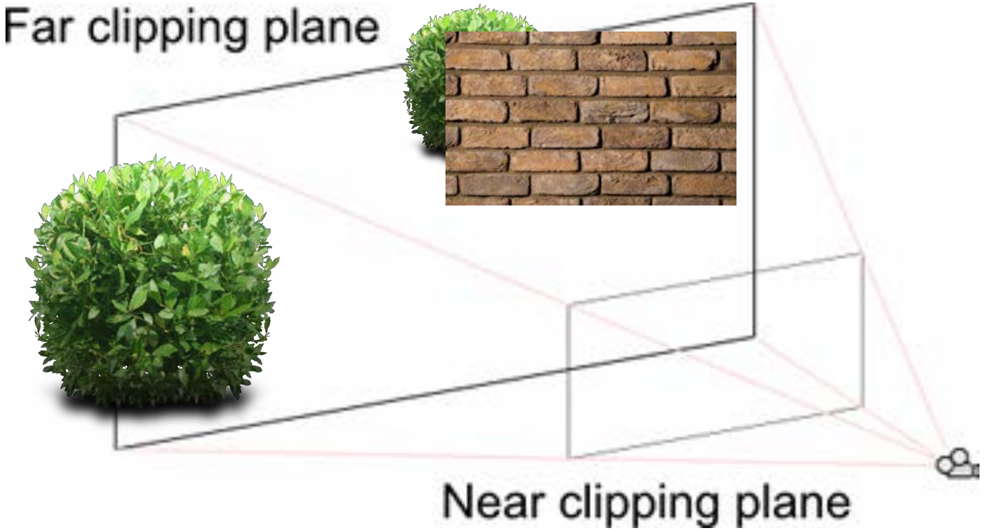


Graphics - Updating the Screen

Occlusion culling problem: don't render hidden objects

Frustum culling: test if an object intersects with the frustum.

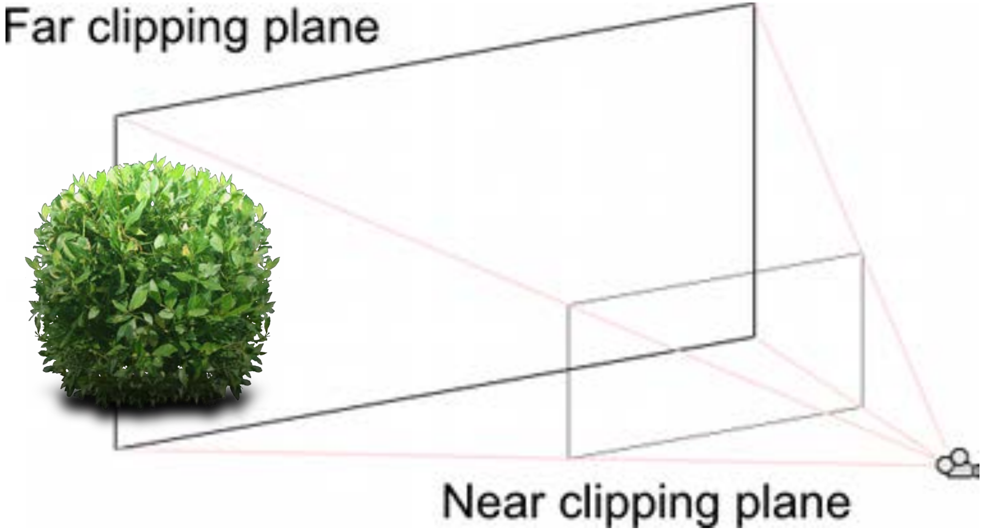
Portals: designers *manually* place simple primitives around chunks of the game world. The portals are invisible but cheap to test intersection on.



Graphics - Updating the Screen

PVS: Potential visibility set precomputed. Very efficient for small environments. PVS is submitted to the renderer and items in the set are tested to make sure they are indeed visible

Bad: storage costs



Art assets & animation

Graphics

Physics engines

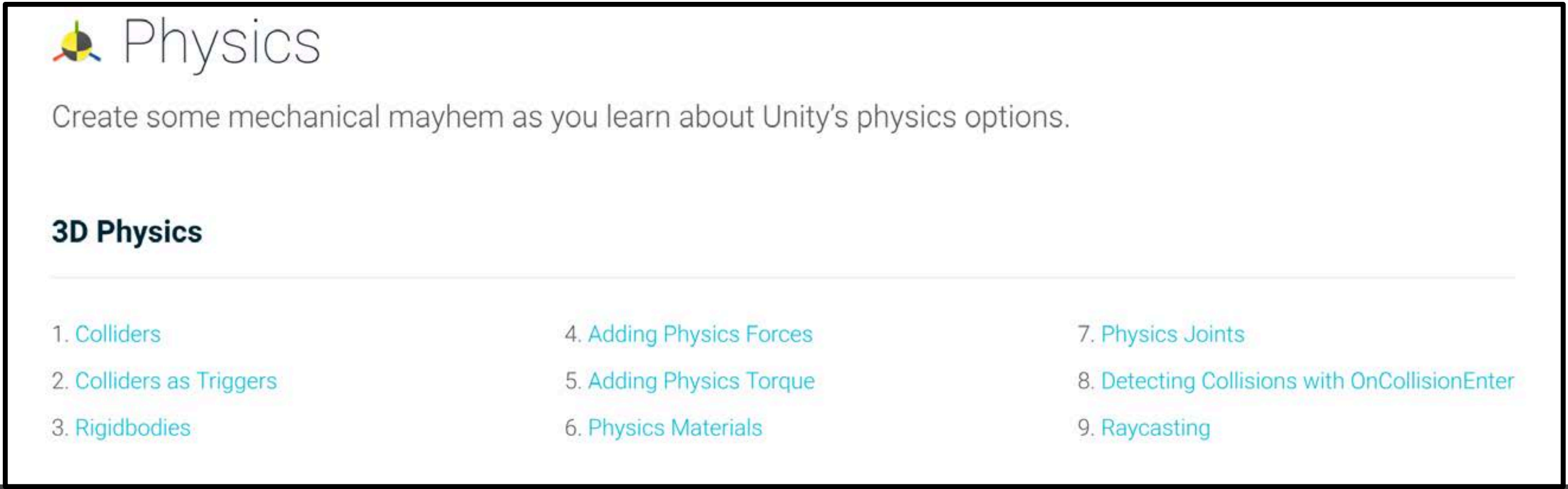
Game loop

Physics

Physics engines and tools

Physics

Unity or Unreal game engines have basic built-in libraries.



Physics

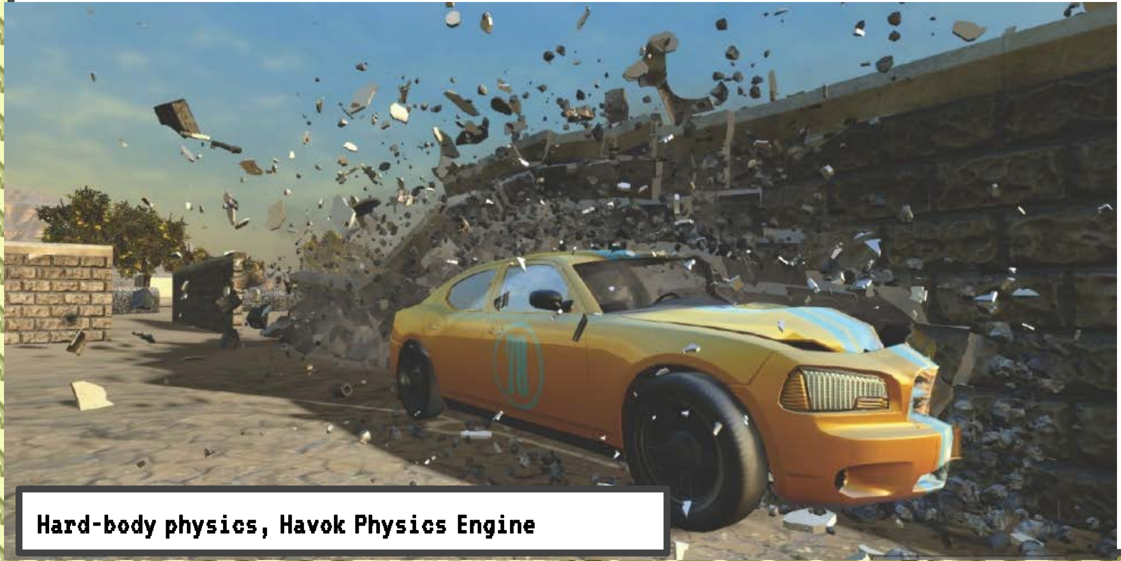
Create some mechanical mayhem as you learn about Unity's physics options.

3D Physics

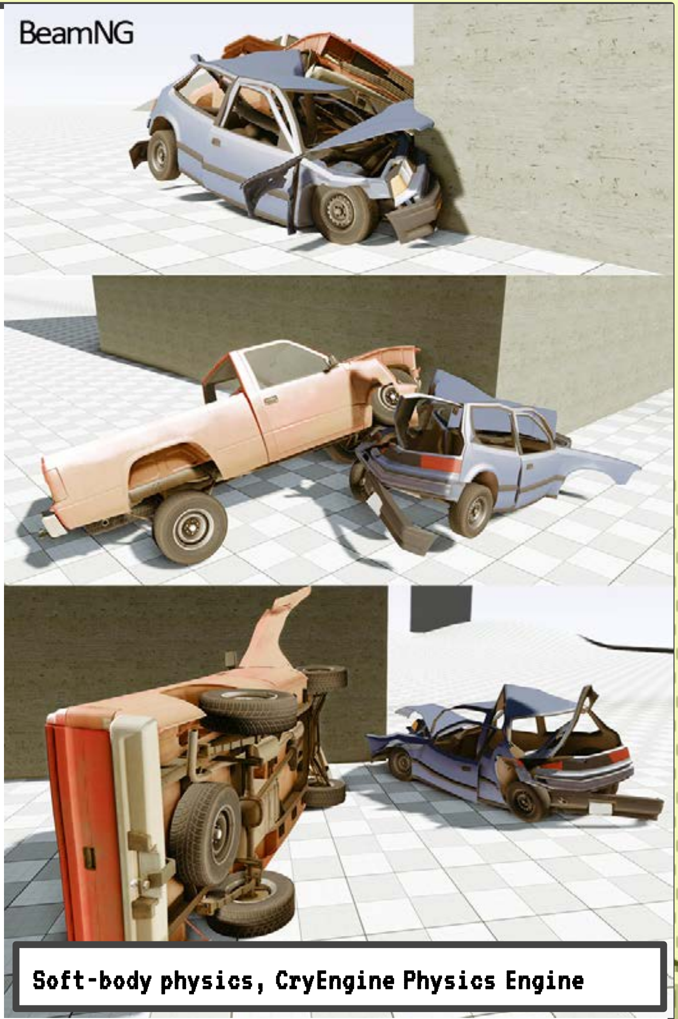
1. Colliders
2. Colliders as Triggers
3. Rigidbodies
4. Adding Physics Forces
5. Adding Physics Torque
6. Physics Materials
7. Physics Joints
8. Detecting Collisions with OnCollisionEnter
9. Raycasting

Physics engines

Calculate on-the-fly physics simulations, optimized for a game environment.



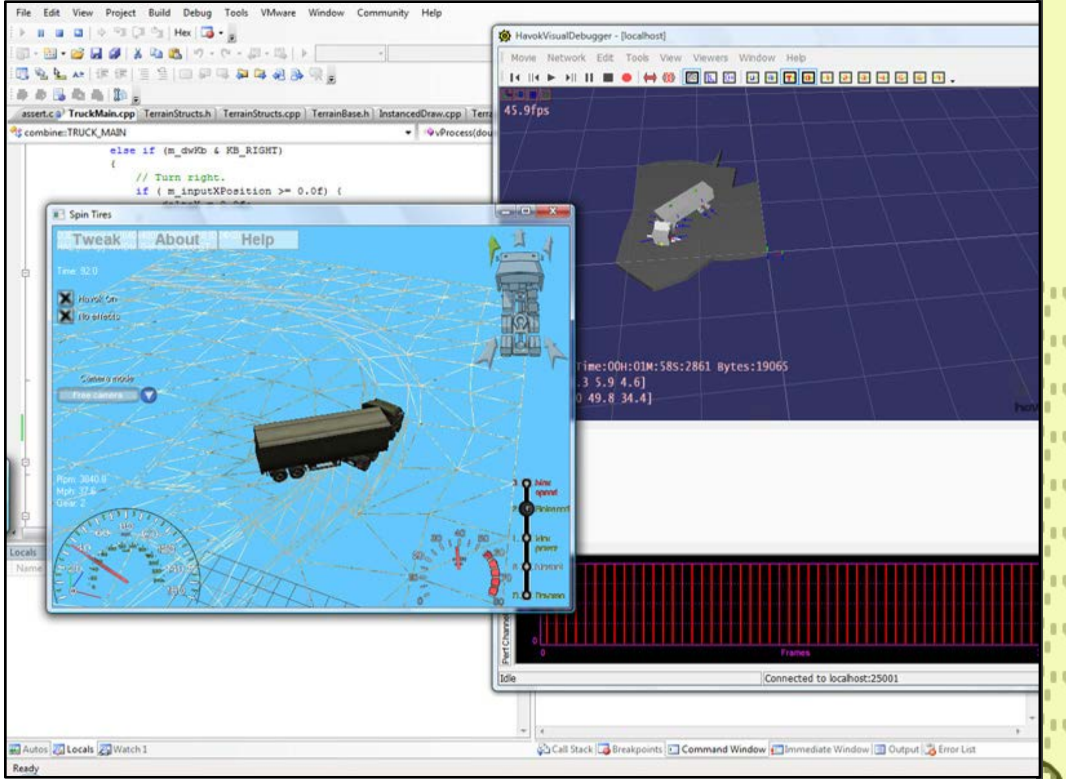
Hard-body physics, Havok Physics Engine



Soft-body physics, CryEngine Physics Engine

Physics engines

SDKs with visual debuggers that allow you to run physics simulations on your object to test your code



```
            else if (m_dwKb & KB_RIGHT)
            {
                // Turn right.
                if ( m_inputXPosition >= 0.0f) {
```

Time: 82.0

☒ Havok On
☒ No effects

Rpm: 3840.8
Mph: 37.6
Gear: 2

```
Time:00H:01M:58S:2861 Bytes:19065
.3 5.9 4.6]
0 49.8 34.4]
```

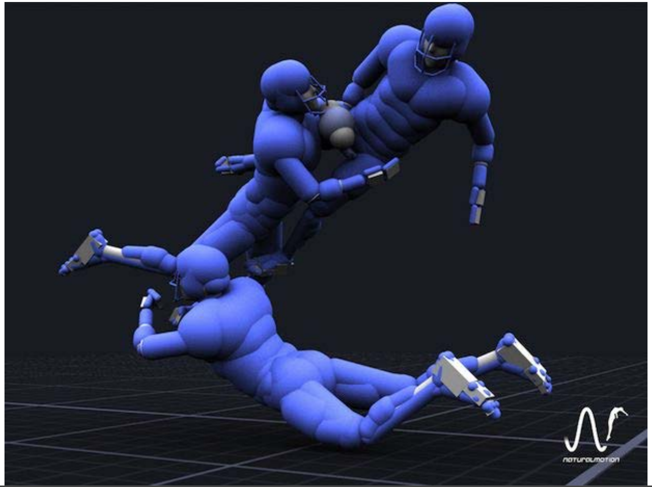
Idle

Connected to localhost:25001

Ready

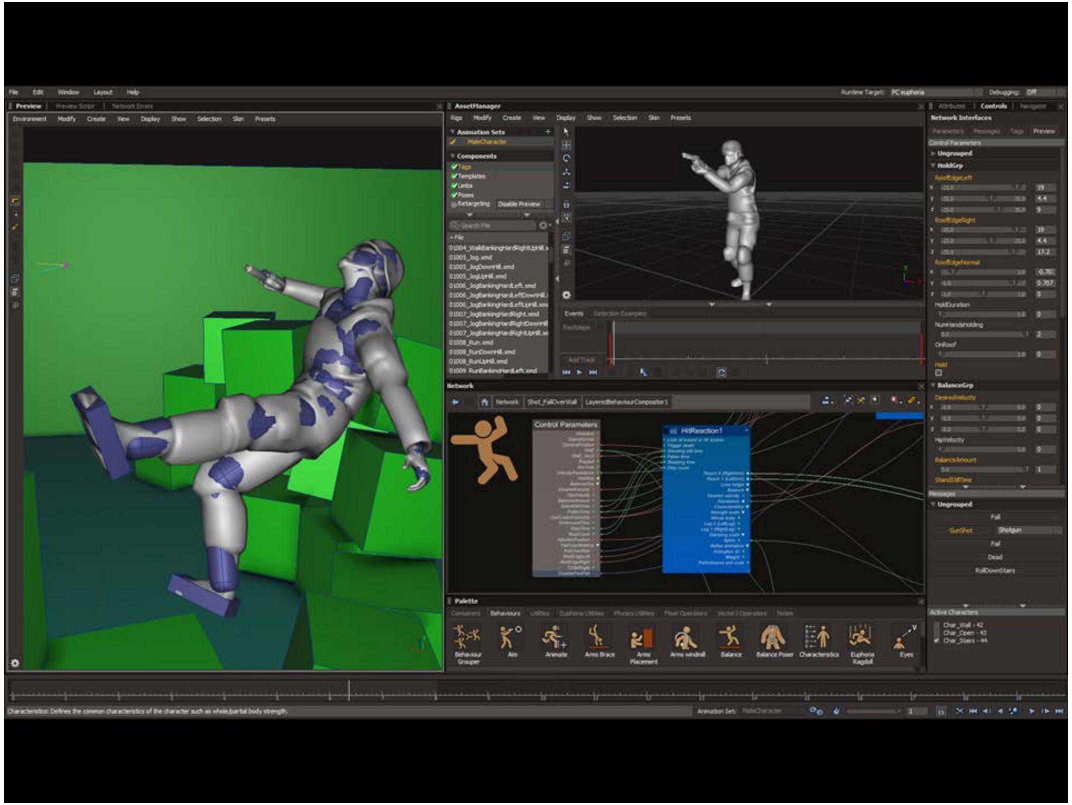
Dynamic animation

Euphoria by Natural Motion encodes lots of information about human muscles, bones, and nerves to dynamically create realistic character movement like falls.



Dynamic animation

Natural Motion editor with visual programming.



Characteristics: Defines the common characteristics of the character such as whole/partal body strength.

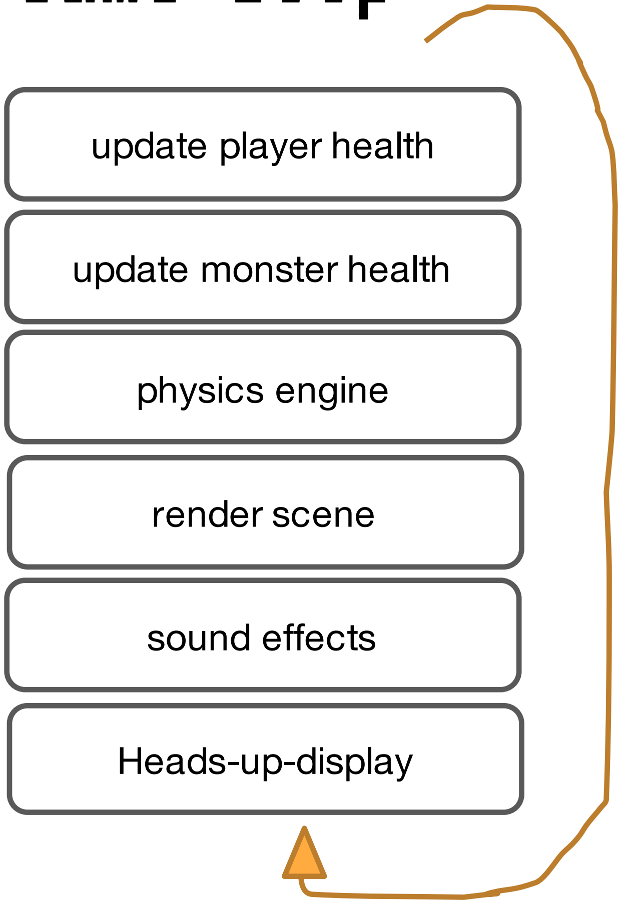
Art assets & animation

Graphics engines

Physics engines

Game loop

Game loop



update player health

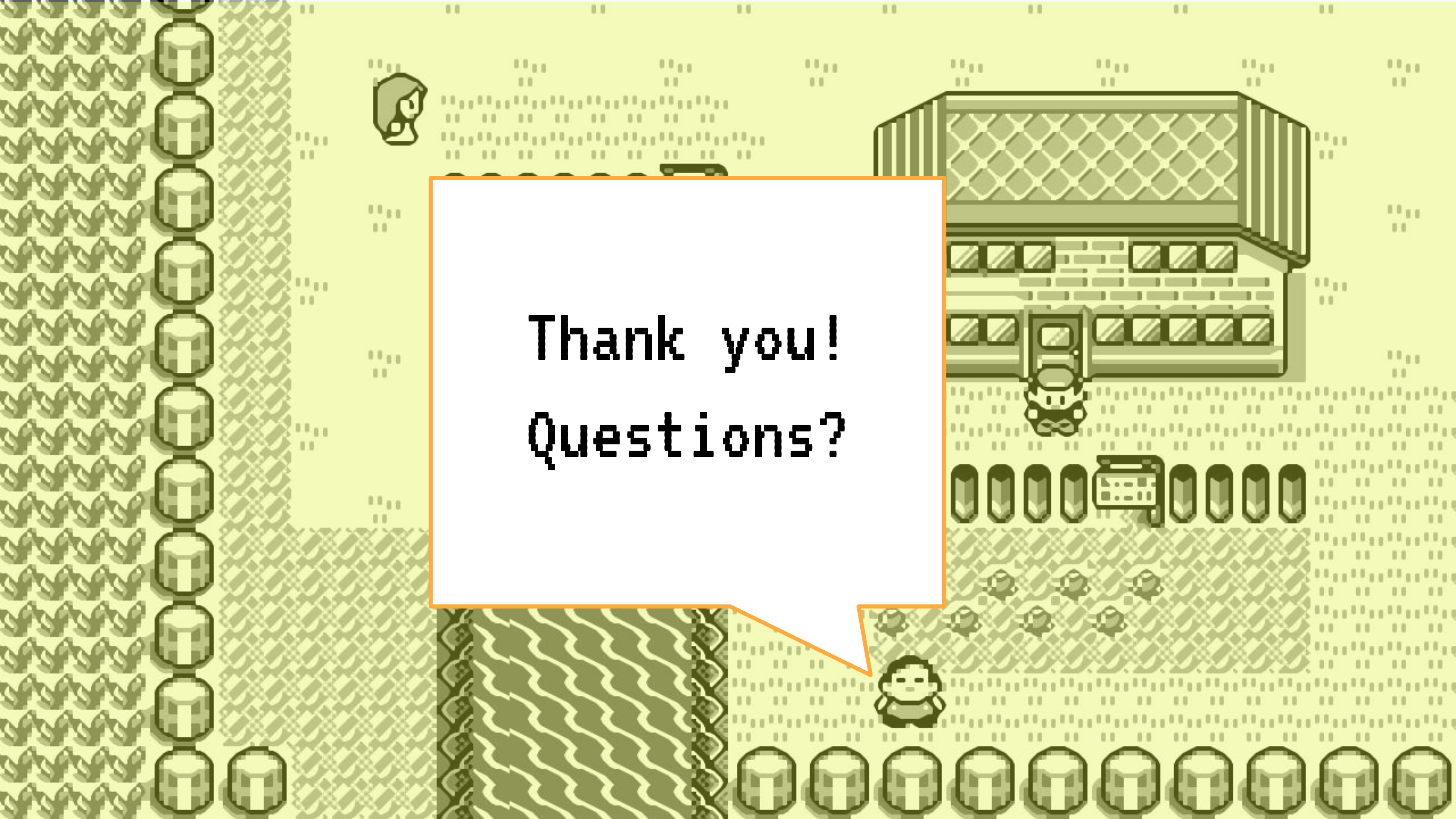
update monster health

physics engine

render scene

sound effects

Heads-up-display



Thank you!

Questions?