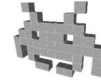
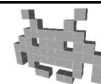


3D Video Games

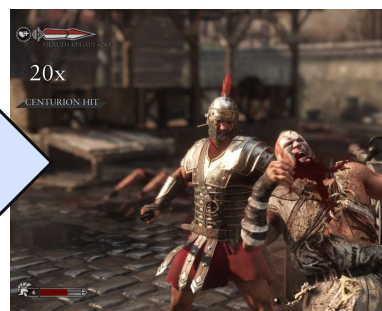
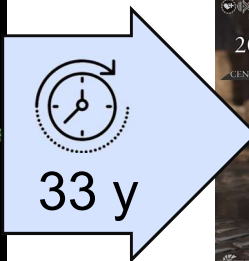


- Core techniques used in modern 3D games
- It's a quite established set of specific methodologies !

3D Video Games

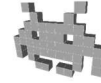


Battlezone – Atari 1980



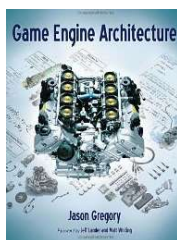
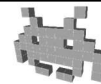
Ryse – Crytek 2013

Corse objectives: the behind the scenes of a game!



- Mathematics for 3D games
- 3D Computer Graphics for games
- 3D Computer Animations for games
- 3D Scenes in games
- 3D Assets in games
- Special Effects in 3D Games (notes)
- 3D Sound in game (connection to other courses)
- Artificial intelligence for 3D games (connection to other courses)

Possible textbooks

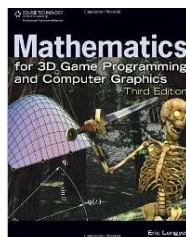


Game Engine Architecture

Jason Gregory

Abb. completo (con cenni di:

software tools, software eng., AI prog, CG prog, math, game design...)



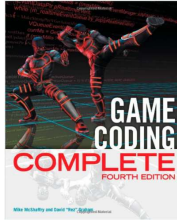
Mathematics for 3D Game Progr. and C.G.

(3za ed)

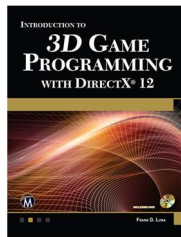
Eric Lengyel

Copre bene il lato + tecnico: 3D math, CG pipeline, geometry + transforms, raytracing, visibility, physic sims, semplice geom processing...

Other relevant books



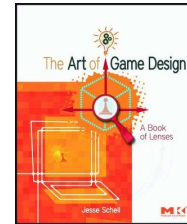
Game Coding Complete (4th ed)
Mike McShaffry, David Graham
Visione pratica
(ma attenzione a ossolescenza)
Accento su coding, software eng
(es memory management).



Introduction to 3D Game Programming with DirectX 12
Frank Luna
Rendering / GPU
(basically, Computer Graphics for games)

The Art of Game Design

Jesse Schell
poco tecnico,
per designers!



Tools which we will adopt

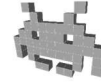
- Existing engine / IDE



OR

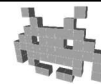


3D Video Games: fun facts



- Huge industry
- Video games = killer apps
- Technology impulse (HW e SW)
- Performance *and* complexity

The academic side conferences on Video Game Dev



- SIGGRAPH
 - ACM Special Interest Group
- i3D
 - Interactive 3D
- GDC
 - Game Developers Conference
- E3
 - Electronic Entertainment Expo
- PAX
 - Penny Arcade Expo



Course Plan



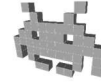
- lec. 1: Introduction ●
- lec. 2: 3D Game Mathematics ●●
- lec. 3: Scene Graph ●●
- lec. 4: Game 3D Models ●●●
- lec. 5: Game Textures ●●
- lec. 6: Game 3D Physics ●●● + ●●●
- lec. 7: Game Particle Systems ●
- lec. 8: Game 3D Animations ●●●●
- lec. 9: Networking for 3D Games ●●
- lec. 10: Artificial Intelligence for 3D Games ●●
- lec. 11: Game 3D Audio ●
- lec. 12: Game 3D Rendering Techniques ●

Categories: according to gameplay



- Puzzle game
 - Color matching
 - Hidden object
 - Trivia game ...
- Action game
 - Beat'em up / hack'n'slash
 - Fighting
 - Pinball
 - Platform
 - Maze
 - Shooter
 - FPS
 - MMO FPS
 - LightGun
 - Shoot'em up (shumps)
 - Rail shooter
 - 3rd person
- Action-Adventure
 - Stealth
 - Survival horror
 - Exploration
 - PoP / Tombrider
- Adventures
 - IF - Interactive Fiction
 - Real time 3D adv
 - Point and click
- Board game
 - Card game ...
- Strategy
 - 4X
 - RTS
 - Strategy MOBA / MMOG
 - Action-RTS
 - Tower defences
- Vehicle simulation
 - Driving simulator
 - Flight simulator
 - Amateur
 - Combat
 - Space ...
 - Racing game
 - Vehicular combat
- Role-playing games
 - RPG (occidentali, orientali)
 - Sandbox RPG
 - MMOPRG
 - Roguelikes
 - Action RPG
- Sport games
 - Soccer / Football / ...
- Simulation / management

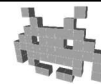
Categories: according to users



casual games vs hard core games



Categories: according to platforms



- Arcade
- PC stand-alones
 - Aka “desktop app”
 - (“computer game” propriamente detto)
 - Win, Mac, Linux...
- Console
 - Wii, PS, Xbox ...
- Browser
 - html5, WebGL, unity, flash...
- Mobile devices
 - Android, iDevices, PSP ...



Categories: according to developer

Independent games

- No/tiny publisher:

Mainstream games

- Big publisher



What does a video-game publisher do?

- Tasks:
 - fund developments
 - including licences
 - distribution
 - marketing
 - ads, launch, market surveys...
 - packaging, manuals
 - localization- High risk



Categories: according to developer

Independent games

- No/small publisher
- Low starting \$
- Small Dev-Teams
- + freedom +novelty
 - (traditionally)
- In need of alternatives for:
 - Funding
 - e.g.: Crowd funding
 - see [indiegogo.com](#), [kickstarters.com](#), ...
 - Distribution
 - e.g.: steam, popcap, apple store...

Mainstream games

- Big publisher
- Big \$ per project
 - (at times, mega-\$'s)
- High quality: a must
- Large Dev-teams

Categories: 2D or 3D?

2D games

- Sprites + Tilemap

3D games

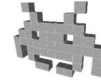
- Modelli + Scene 3D

TileSet

TileMap

Sprites

Categories: 2D or 3D?

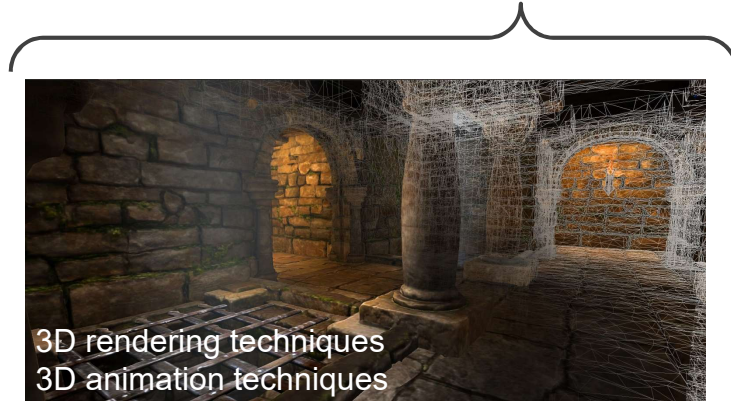


2D games

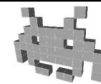
- Sprites + Tilemap

3D games

- 3D Models + 3D Scenes



Categories: 2D or 3D?



2D games

- Sprites + Tilemap
- Techniques:
 - Blitting
 - Tilemaps
 - and 2D scrolling
 - Sprite support
 - sprite collision-detection
 - 2D transform
 - (2D physical engines)




3D games



- 3D models + 3D Scenes
- Techniques :
 - 3D Modelling
 - Scenegraph, models
 - 3D Real time rendering
 - 3D transform
 - lighting
 - 3D animations
 - Kinematics, motion capture, model animations...
 - 3D physical simulations
 - 3D sound localization

Categories: 2D or 3D?

2D games




- Sprites + Tilemap
- Tools:




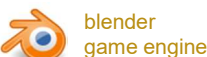


GameMaker: Studio


...

3D games

- 3D Models + 3D Scenes
- Tools:


unity
OGRE


...

Note: we are interested in the tech not the gameplay

2D tech



3D tech



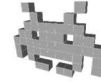
2D gameplay



3D gameplay



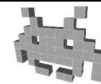
3D VideoGames







- Common tasks need be faced
 - 3D Rendering
 - Real time transform + lighting
 - 3D Physics
 - Newtonian physical simulations
 - Collision detection + response
 - Networked 3D Physics
 - 3D Sound rendering
 - Input management
 - Program structure
 - event loop
 - Memory management
 - Artificial intelligence
 - many common sub-task (in a 3D scene)
 - goal-subgoal hierarchy
- Animations
scripted or computed

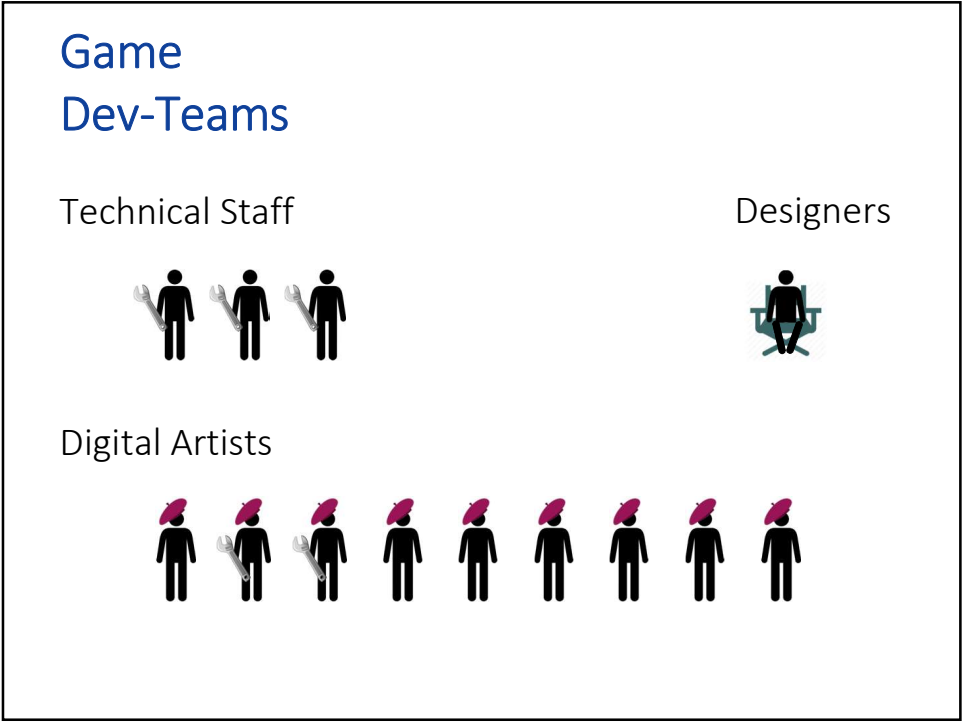
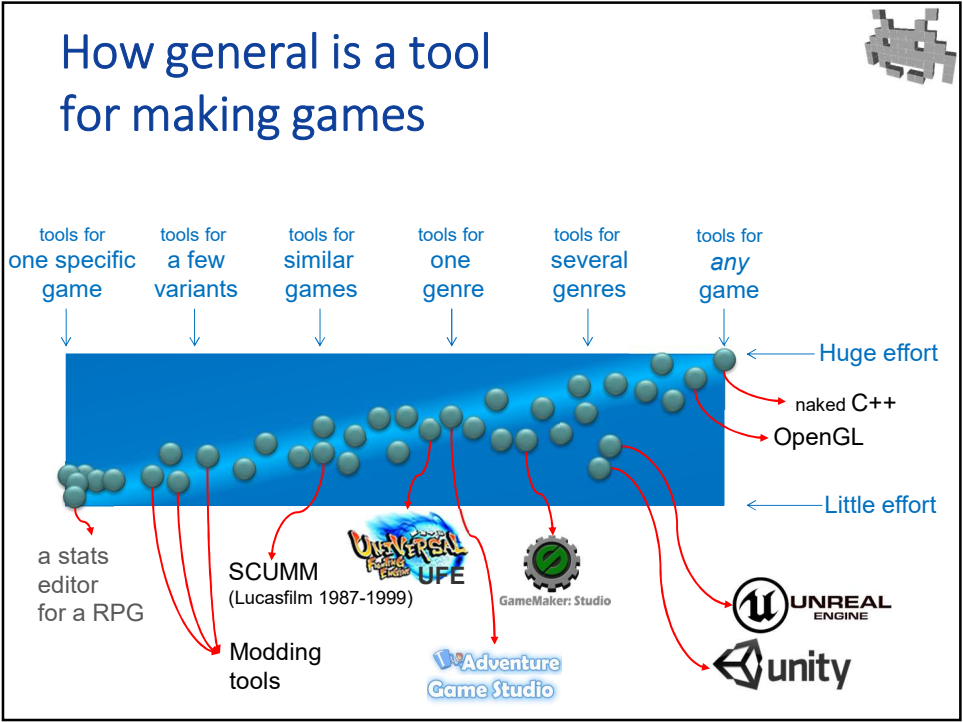
REUSE!

Implement once, use many times



- Still possible to make games completely from scratch (zero reuse), but increasingly rare.
 - Even many projects/series started this way then switch to a game engine
- Game-engines take care of many common functionalities needed by different games.
 - eg:





- But
 - Reuse = constraints
 - Zero reuse → maximal freedom

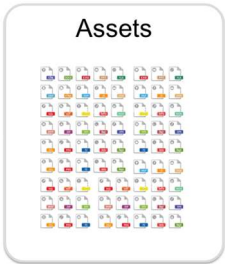


Game Dev-Teams

Technical Staff

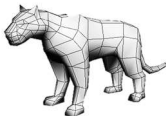


Digital Artists



Game assets! (aka game contents)

- 3D data
 - models
 - textures
 - materials
 - shaders
 - animations
 - collision objects
 - scenes
 - etc



- audio
 - music
 - sound fxs
 - ambient sounds
 - voice overs
 - etc



- video
 - cut-scenes, intros, etc



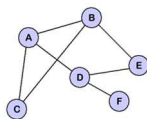
- 2D art
 - screen splashes
 - backgrounds
 - GUI / HUD elements
 - [sprites & tile-sets ?]
 - fonts
 - etc

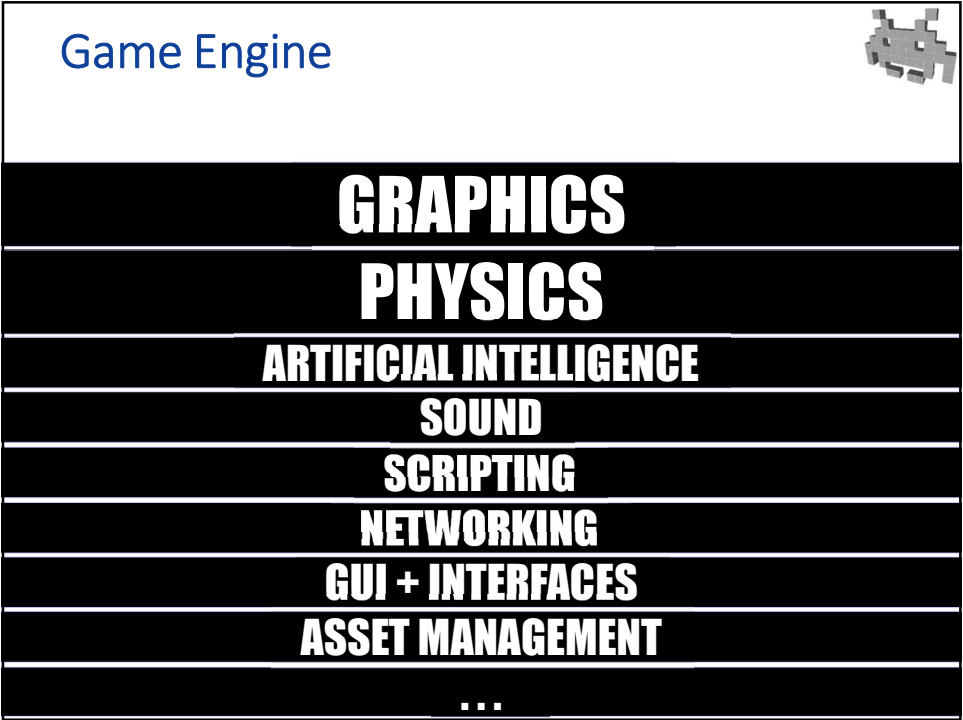
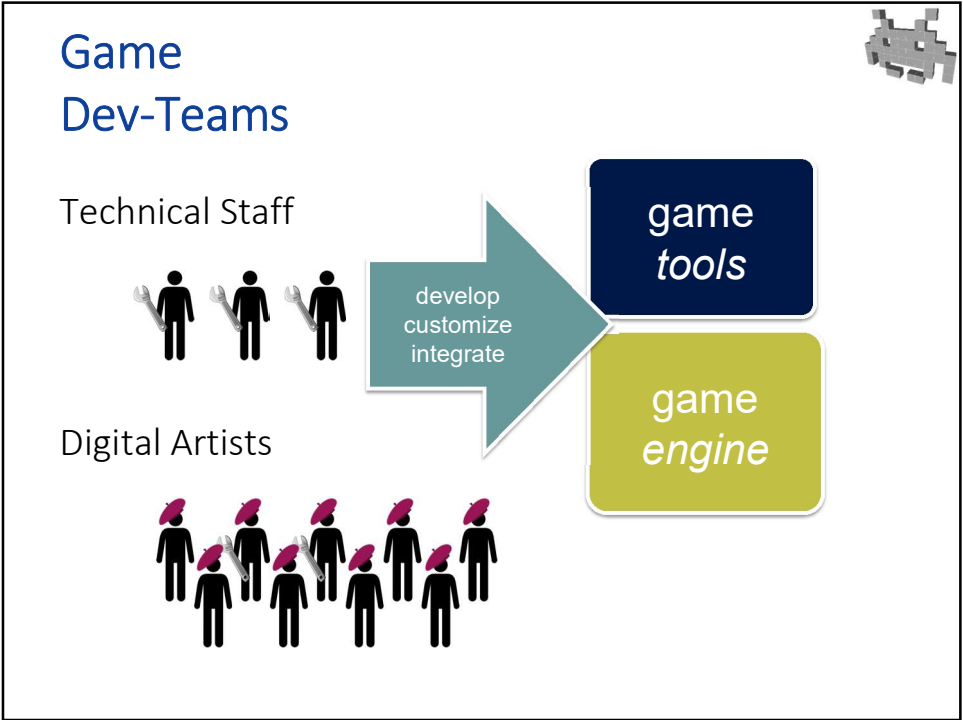


- text
 - dialogues trees
 - messages
 - translations
 - etc



- etc:
 - scripts
 - stats
 - levels
 - etc

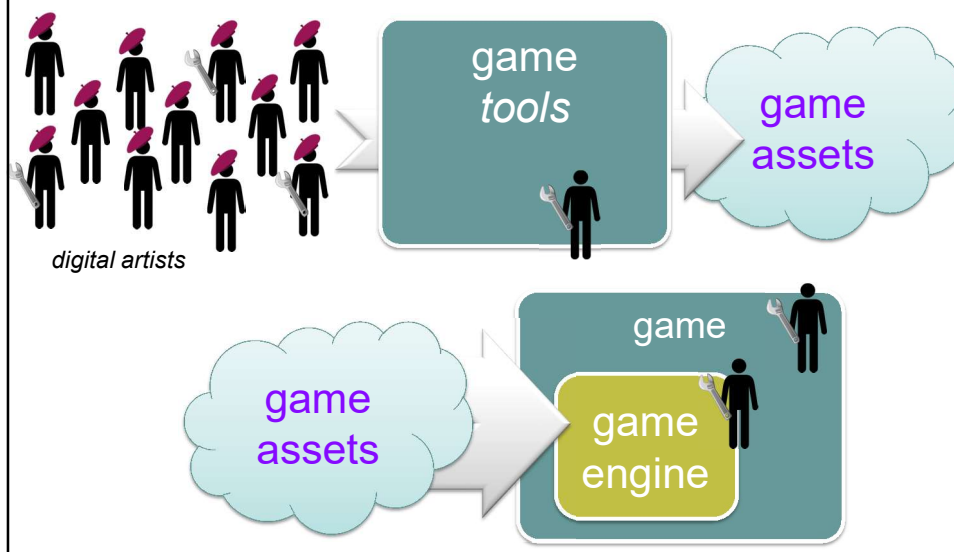


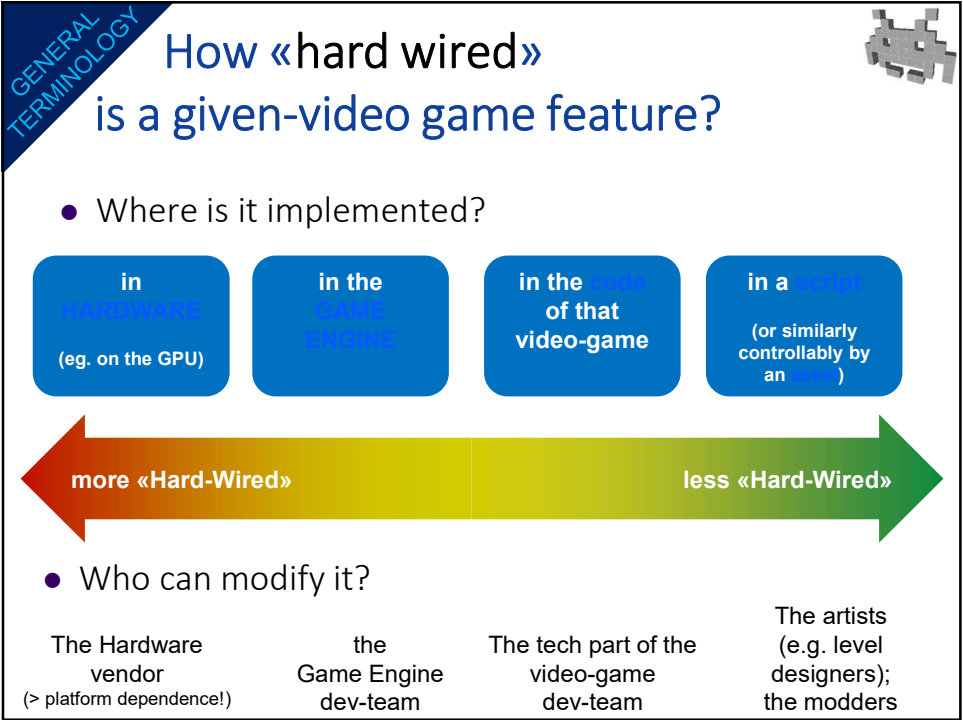
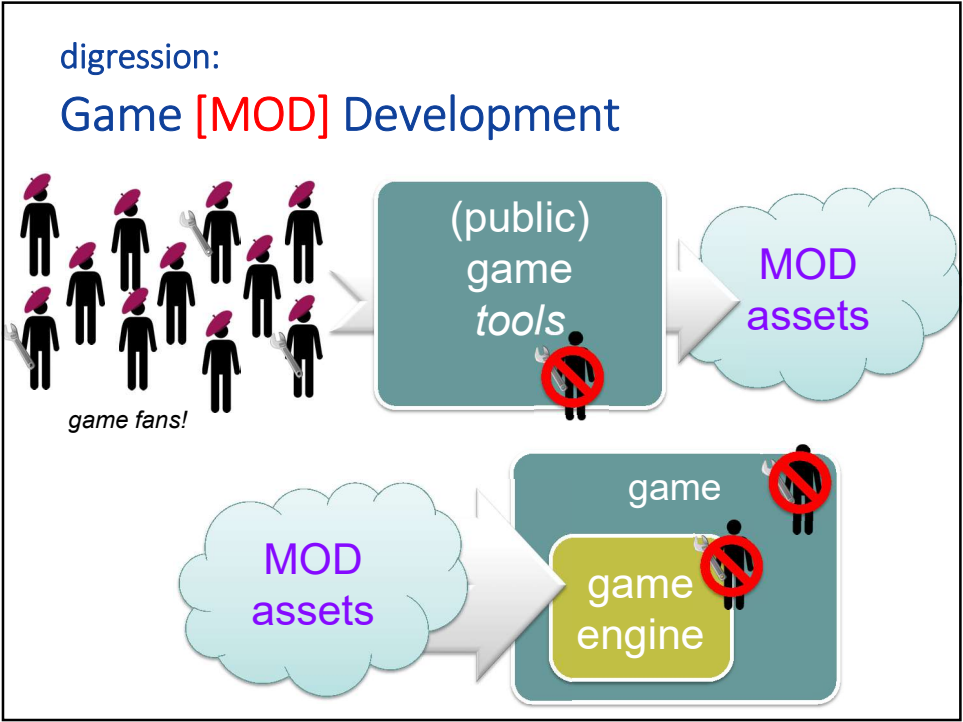


Game Engine

- Part of game SW which deals with a set of common tasks:
 - Handling of the 3D Scene
 - Renderer
 - Real time transform + lighting
 - Models, materials ...
 - Physics engine
 - (soft real-time) newtonian physical simulations
 - Collision detection + response
 - Networking
 - (LAN – eg via UTP)
 - “Sound-renderer”, Sound mixer
 - Unified handling of HCI devices
 - Main event loop, timers, windows manager...
 - Memory management
 - AI module
 - Solutions to many common AI task
 - Localization support
 - Scripting
 - GUI (HUD)
- } Animations
scripted or computed


Assets in Game Development





GENERAL TERMINOLOGY


How «hard wired» is a given-video game feature?



More Hard-wired	Less hard-wired
<ul style="list-style-type: none"> • > efficiency • > scalability • > reuse 	<ul style="list-style-type: none"> • > ease of maintenance • > customizability • > flexibility

GENERAL TERMINOLOGY

A general concept we will be encountering it several times



ASSET - STORED	VS	PROCEDURAL - GENERATED
<ul style="list-style-type: none"> • Build during the dev of a game <ul style="list-style-type: none"> • « it is designed » • « it is scripted » • > quality (usually) <ul style="list-style-type: none"> • (if the artists are good) • > control <ul style="list-style-type: none"> • by the digital artist • (time efficiency <small>maybe</small>) 		<ul style="list-style-type: none"> • Express made during game execution <ul style="list-style-type: none"> • « it's a procedure » • « it's dynamically computed » • > variations <ul style="list-style-type: none"> • which is tied to "replayability" • > flexibility <ul style="list-style-type: none"> • (can adapt to the context) • (space efficiency <small>RAM, DISK...</small>)

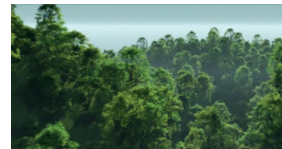
Procedural generation In games

For example

- Procedural levels / missions
- Procedural Terrain
- Procedural AI
- Procedural «Bosses»
- Procedural Scenes
- Procedural Models
- Procedural Textures
- Procedural Animations (physics)
- Procedural Music ...



Rogue, Michael Toy et al, 1980



Procedural Forest in ICE



Un roguelike



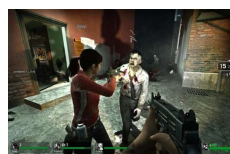
Shadow Over Mordor, Monolith Prod., 2011



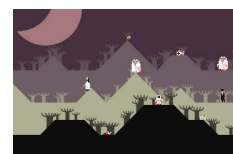
Minecraft, Mojang, 2009



Elite, Acornsoft, 1984



Left 4 dead, Valve, 2008




Rescue the beagles, 16x16, 2008

GENERAL
TERMINOLOGY

«Baking», «Baked» / «Pre-baked»

it: "cuocere (al forno)"



definitely, as one **asset**
(otherwise, it's **caching**)

baking : **Storing for good** the result of
a **computation**, for later use

note: several assets are juts
baked-*something* (e.g. a baked light-map,
a baked animation) or have baked elements

often, (refined versions of)
the ones normally employed in **real time**

We gain:

- **time** (CPU / GPU workload)
- almost total independence from computation complexity !
→ less compromises,
more quality

We pay with:

- **space**
(on **disk** , **Ram** , **GPU RAM**)
- loss of **flexibility**
(all the parameters used by the
computation are frozen)


Game assets!

(aka game contents)


- 3D data
 - models
 - textures
 - materials
 - shaders
 - animations
 - collision objects
 - scenes
 - etc
- audio
 - music
 - sound fxs
 - ambient sounds
 - voice overs
 - etc
- video
 - baked cut-scenes, intros, etc

- 2D art
 - screen splashes
 - backgrounds
 - GUI / HUD elements
 - [sprites & tile-sets ?]
 - fonts
 - etc
- text
 - dialogues trees
 - messages
 - translations
 - etc
- etc:
 - scripts
 - stats
 - levels
 - etc


The 3D part of game assets



- 3D Models**
i.e. tri-meshes with:
 - per vertex attrib
 - normals, color, AO, ...
 - LODs
 - "uv-mapping"
 - keyframes
 - cyclic animations
 - face-morphs, ...
 - "skinning"



- Materials**
 - lighting model stats / flags
 - textures
 - RGB maps
 - normal maps
 - alpha maps ...
 - shaders
 - vertex, fragments, ...



- Animations**
 - blend shapes
 - skeletal animations
 - kinematic animations
 - geometry caches
- Geometric proxies**
 - hit-boxes
 - bounding objects
 - AI-meshes
- Particle systems**
 - (represented by a firework image)
- Environments**
 - scene-graphs
 - skydomes
 - 2.5D terrains