

# 3D Video Games

Marco Tarini

Università degli Studi di Milano

2022/2023

- Core techniques used in modern 3D games
- A well-established set of specific methodologies used in most 3D games

2

# Game 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 (eastern, western)
  - Sandbox RPG
  - MMOPRG
  - Roguelikes
  - Action RPG
- Sport games
  - Soccer / Football / ...
- Simulation / management

4

## Categories: according to player types

casual games

vs

hard core games



5

## Categories: according to platforms

- Arcade
- PC stand-alones
  - Aka "desktop app"
  - Win, Mac, Linux...
- Console
  - Wii, PS, XBox ...
- Browser: game = web app
  - html5, WebGL, unity, flash...
- Mobile devices
  - Android, iDevices, PSP ...



6

## Categories: according to developer

### Independent games

- No/tiny publisher:

### Mainstream games

- Big publisher

Logos for various game publishers and developers, categorized into independent and mainstream.

7

## What does a video-game publisher do?

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

High risk!

+++

ACCOLADE namco

Broderbund

8

Marco Tarini  
Unviersità degli studi di Milano

3

### 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



9

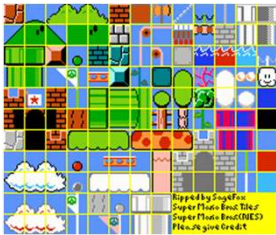
### Categories: 2D or 3D?

#### 2D games

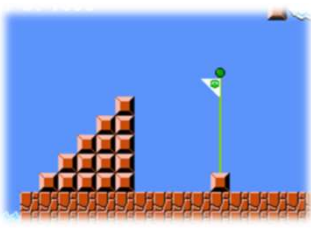
- Sprites + Tilemap

#### 3D games

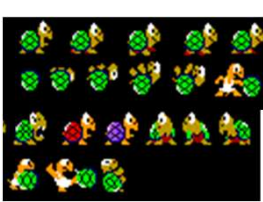
- 3D Models + 3D Scenes



TileSet



TileMap



Sprites

10

## Categories: 2D or 3D?

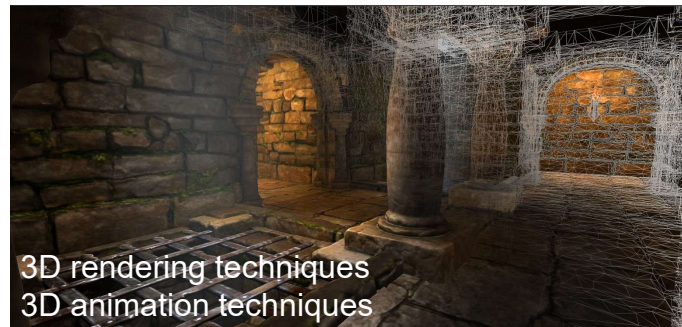


### 2D games

- Sprites + Tilemap

### 3D games

- 3D Models + 3D Scenes



11

## 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

12

### Categories: 2D or 3D?












#### 2D games

- Sprites + Tilemap
- Tools:



#### 3D games

- 3D Models + 3D Scenes
- Tools:



13

### Note: we are interested in the tech not the gameplay

#### 2D tech

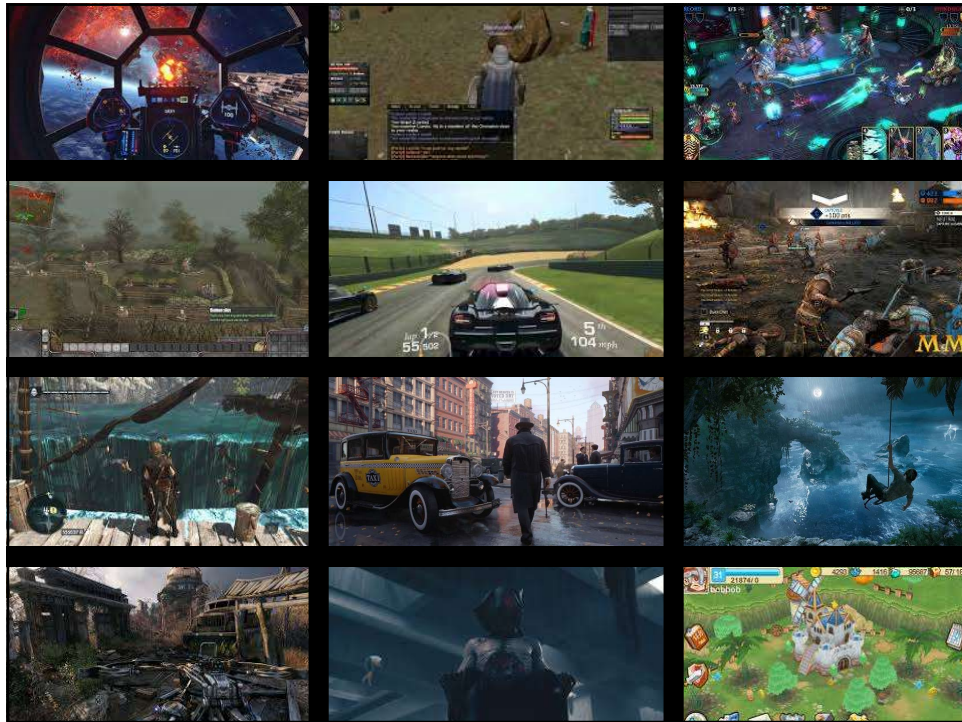


#### 3D tech



14





15

## About this course: webpage

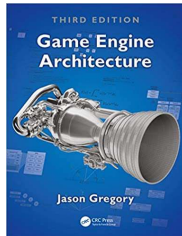


- Follow the link from [Ariel](#)
- or
  - Search for my name: [Marco Tarini](#)
  - Land on my [unimi](#) page
  - Follow [3D Videogame](#) link
- or  
<https://tarini.di.unimi.it/teaching/3DVG2023/>

18

## About this course:

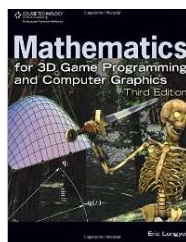
### Potentially useful textbooks



#### Game Engine Architecture

Jason Gregory

Complete (notes on:  
software tools, software eng., AI prog, CG prog, math, game  
design...)



#### Mathematics for 3D Game Programming and C.G.

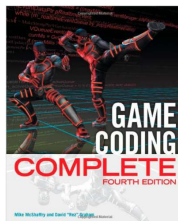
(3rd ed)

Eric Lengyel

Good coverage of 3D math,  
(and, CG pipeline, geometry + transforms, raytracing, visibility,  
physic sims, semple geom processing...)

19

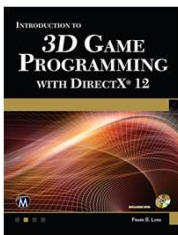
## Other relevant books



#### Game Coding Complete (4th ed)

Mike McShaffry, David Graham

Practical approach  
(sometimes not fully up to date)  
Stress on coding asoect, software eng  
(e.g. memory managment).



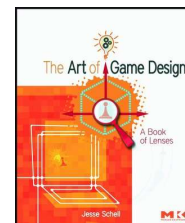
#### Introduction to 3D Game Programming with DirectX 12

Frank Luna

Rendering / GPU  
(basically, Computer Graphics  
for games)

#### The Art of Game Design

Jesse Schell  
not technical,  
focus on design!



20



## About this course: the “game of the week”



After every Monday lecture  
(including today)

- Completely optional
- Not part of grading
  - No extra point
- Not an official part of the course in any sense or form
- Just an occasion to have fun


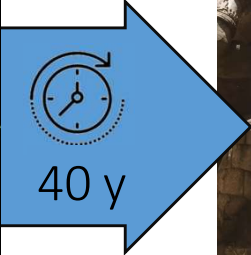

21

## About this course: the exam

- Preliminary Written Test
  - Moodle
  - Closed and short open questions
  - Mini-problems
  - Definitions.
- Oral Exam
  - Covers the entire lectures
  - Procedure: I roll a die, 1-24  
Ask about respective lecture

22

### 3D Video Games



Battlezone – Atari 1980

Unreal Engine V – 2020

26

### Course Plan


- lec. 1: Introduction ●
- lec. 2: Mathematics for 3D Games ●●●●●●●●
- lec. 3: Scene Graph ●●
- lec. 4: Game 3D Physics ●●●● + ●●●
- lec. 5: Game Particle Systems ●
- lec. 6: Game 3D Models ●●
- lec. 7: Game Textures ●●
- lec. 9: Game Materials ●
- lec. 8: Game 3D Animations ●●●●
- lec. 10: Networking for 3D Games ●
- lec. 11: 3D Audio for 3D Games ●
- lec. 12: Rendering Techniques for 3D Games ●
- lec. 13: Artificial Intelligence for 3D Games ●

2h

29

Course Plan

☆ = 8h



lec. 1: Introduction ●

lec. 2: Mathematics for 3D Games ●●●●●●

lec. 3: Scene Graph ●●

lec. 4: Game 3D Physics ●●●●●●

lec. 5: Game Particle Systems ●

lec. 6: Game 3D Models ●●

lec. 7: Game Textures ●●

lec. 9: Game Materials ●

lec. 8: Game 3D Animations ●●●●

lec. 10: Networking for 3D Games ●

lec. 11: 3D Audio for 3D Games ●

lec. 12: Rendering Techniques for 3D Games ●

lec. 13: Artificial Intelligence for 3D Games ●


★ ★  
bases

★ ★  
computer  
animation

★  
“bridge”  
lectures

30

Game Engine: tasks



GRAPHICS

PHYSICS

ARTIFICIAL INTELLIGENCE

SOUND

SCRIPTING

NETWORKING

GUI + INTERFACES

ASSET MANAGEMENT

...

35


## Game Engine

- A game SW suite 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
      - e.g., LAN via UTP...
    - 3D Sound-rendering, Sound mixer
    - Handling of input devices
    - Main event loop, timers, windows manager...
    - Memory management
    - AI module
      - Common solutions to many common AI sub-problem, e.g., routing
    - Localization support
    - Running scripts
    - GUI (e.g., via interactive HUD elements)
- Animations  
scripted or computed

37



## 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

39


### Engines which we will *occasionally* refer or adopt for demonstration

OR


41

### Game Dev-Teams


#### Technical Staff



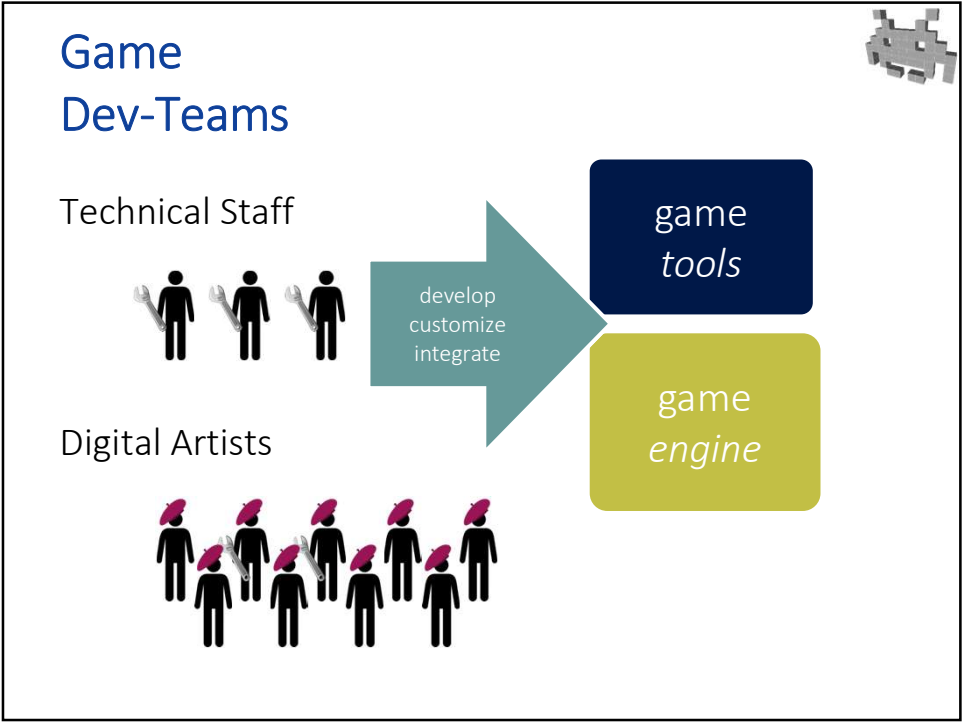
#### Designers



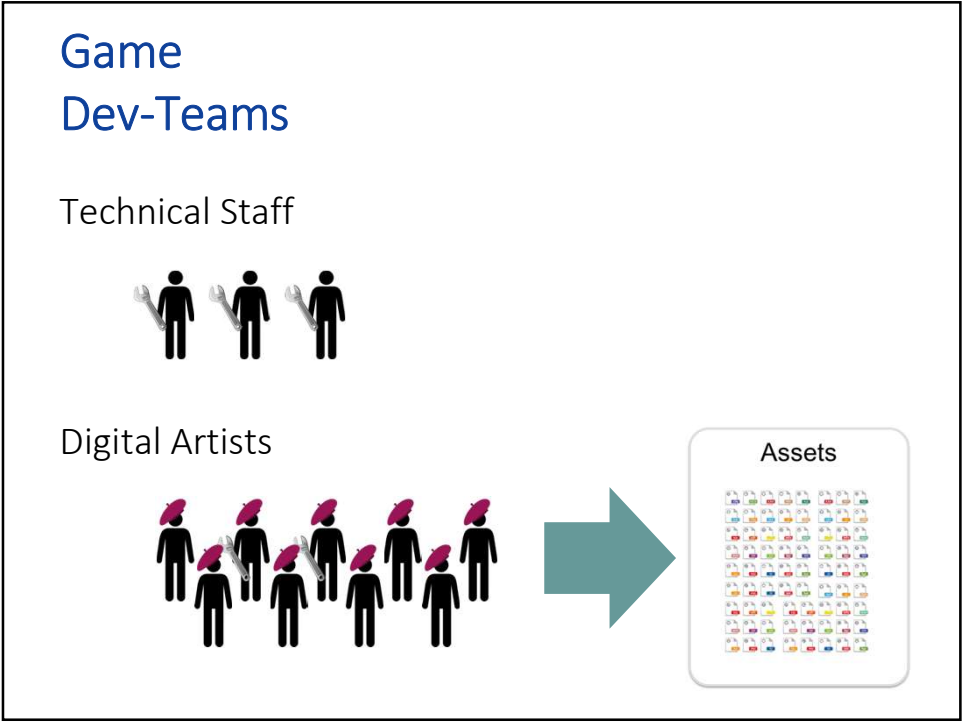
#### Digital Artists



42



43





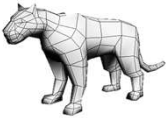
45



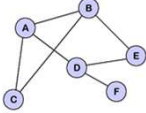


# Game assets

(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



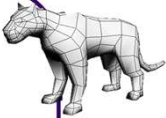


46

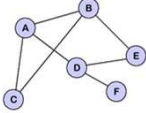


# Game assets

(aka game contents)

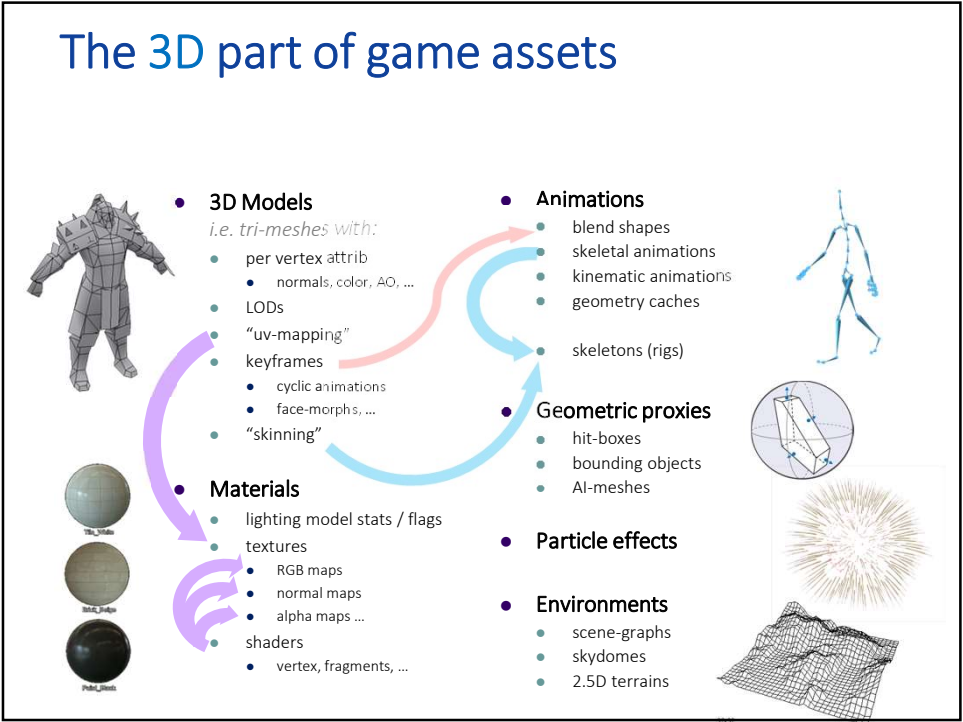
- 3D data
  - models
  - textures
  - materials
  - shaders
  - animations
  - collision objects
  - scenes
  - etc
- audio
  - music
  - sound fxs
  - ambient sounds
  - voice overs
  - etc
- video
  - baked cut-scenes, intros, etc



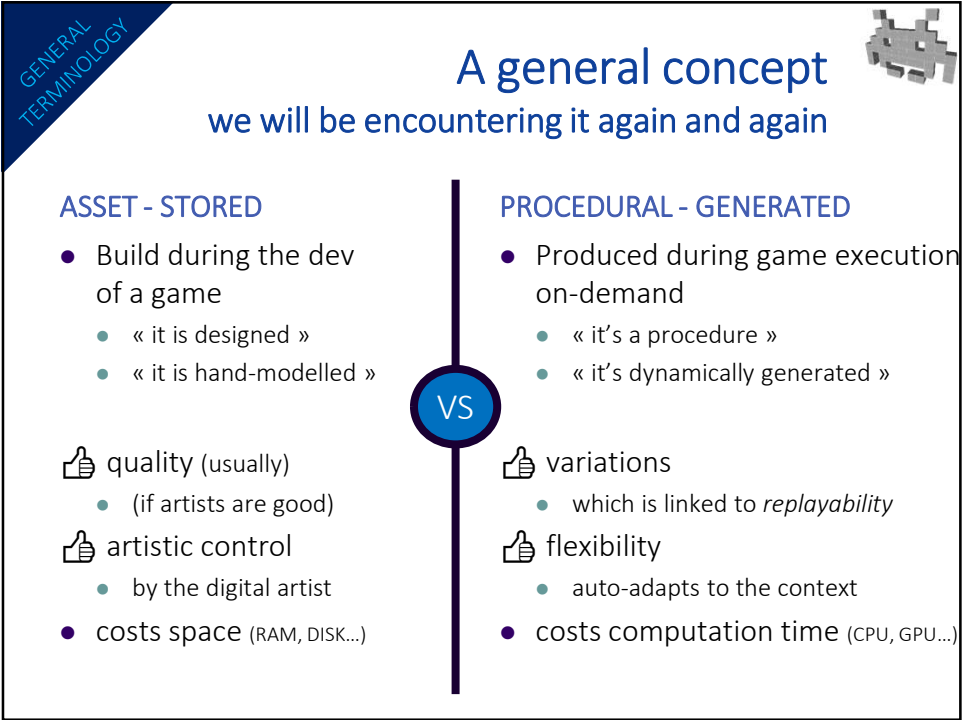
- 2D art
  - screen splashes
  - backgrounds
  - GUI / HUD elements
  - [ sprites & tile-sets ? ]
  - fonts
  - etc
- text
  - dialogues trees
  - messages
  - translations
  - etc
- etc:
  - scripts
  - stats
  - levels
  - etc



47



48




49

# Procedural generation

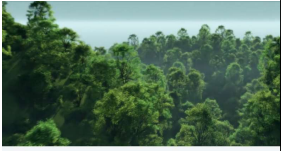
## In games

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




a roguelike




Shadow Over Mordor, Monolith Prod., 2014



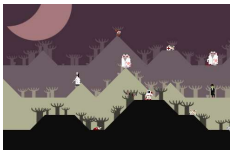
Minecraft, Mojang, 2009



Elite, Acornsoft, 1984



Left 4 dead, Valve, 2008



Rescue the beagles, 16x16, 2008

50

GENERAL TERMINOLOGY

# «Baking», «Baked» / «Pre-baked»

it: “cuocere (al forno)”

once and for all, producing one **asset** (otherwise, it’s **caching**)

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

e.g.: a “baked light-map”, a “baked animation”...

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

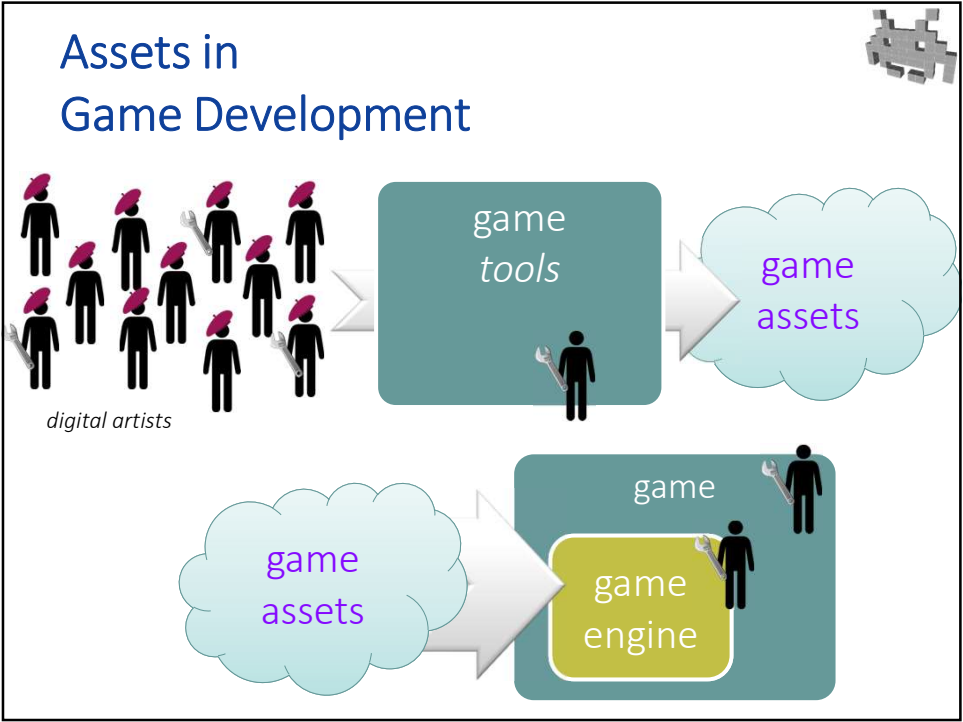
**We gain:**

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

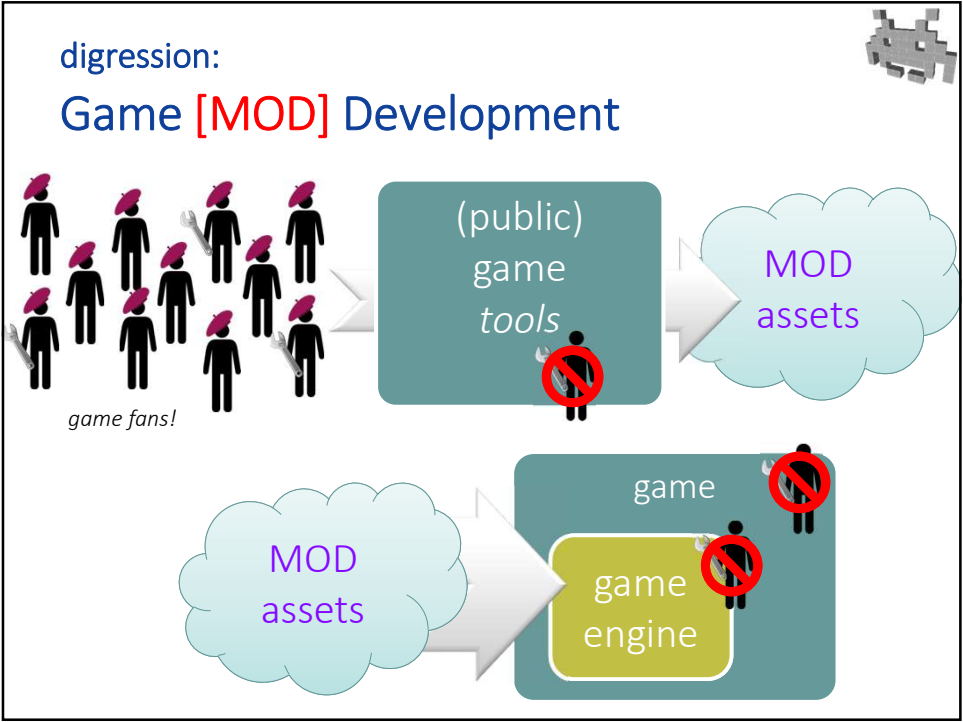
**We pay with:**

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

51



52



53

GENERAL  
TERMINOLOGY

How «hard-wired» (or «hard-coded») is a given-video game feature?

- Where is it implemented?

in  
HARDWARE  
(eg. on the GPU)

in the  
GAME ENGINE

in the **code**  
of that  
video-game

in a **script**  
  
(or similarly  
controllably  
by an **asset**)

more «Hard-Wired»

less «Hard-Wired»

- Who can modify it?

The Hardware  
vendor  
(> platform dependence!)

the  
Game Engine  
dev-team

the tech part of the  
video-game  
dev-team

the artists  
(e.g. level designers);  
the modders

54

GENERAL  
TERMINOLOGY

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

More Hard-wired

👍 more efficiency

👍 more scalability

👍 more reuse

Less hard-wired

👍 ease of maintenance

👍 more customizability

👍 more flexibility

55

Marco Tarini  
Unviersità degli studi di Milano

19