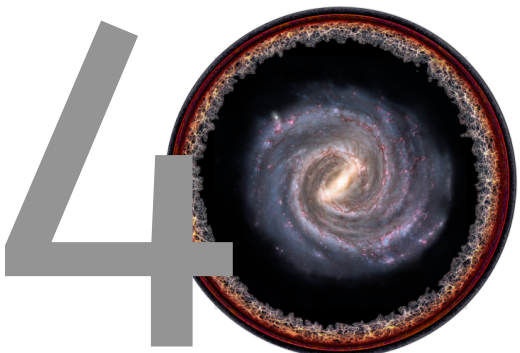
Unity as a Platform

Hierarchy, Memory, ECS

Tomáš Polášek ipolasek@fit.vutbr.cz

Brno University of Technology

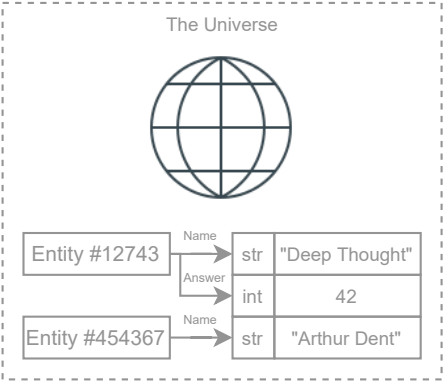
Faculty of Information Technology
DCGM, cPhoto@FIT
Faculty of Fine Arts
Game Media Studio



The Hierarchy

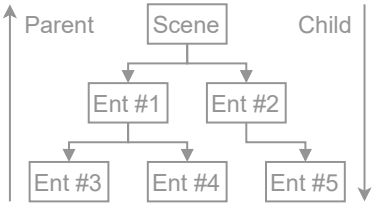
Game World

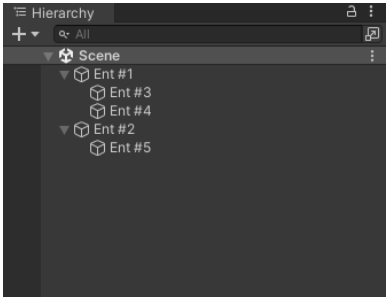
- Virtual Universe = Entities + Data + Structure
  - Entity → Unique Existence
  - Data → Information without Meaning
  - Structure → Meaning and Logic
- Attributes, Properties
- Relations & Interactions



The Hierarchy

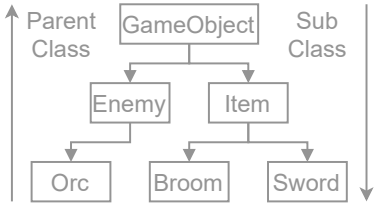
- Scene Hierarchy
- Graph → Tree Structure
- Nodes = Entities
- Links = Hierarchy
- Parent-Child Relationship





The Other Hierarchy

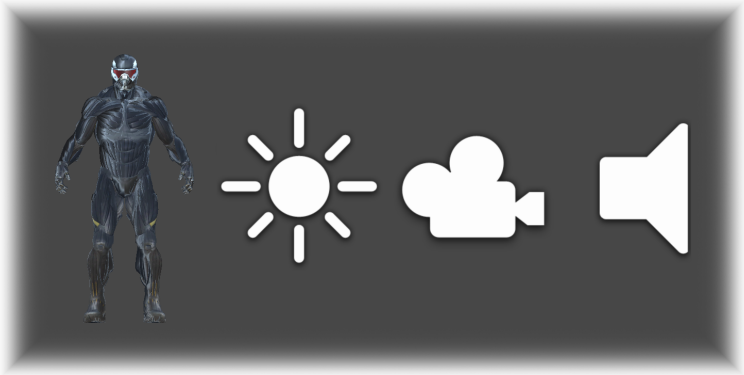
- Hierarchy of Objects
- Non-Cyclic Graph
- Nodes = (Proto) Types
- Links = “Is a” Relationship




```
class GameObject{ /* ... */ }

class Enemy : GameObject
{ int mHealth; }

class Item : GameObject
{ int mCost; }

class Orc : Enemy
{ Orc() { mHealth = 12; } }

class Sword : Item
{ Sword() { mCost = 17; } }

class Broom : Item
{ Broom() { mCost = 12; } }
```

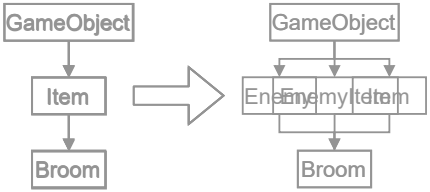
The GameObject

- GameObject = Base for All Game Objects [1, 2, 3, 5]
- Hierarchy → **Behavior** and **Data**
- **Inheritance** × **Composition**



Hierarchy Through Inheritance

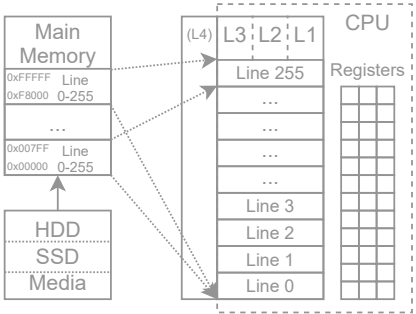
- Class Hierarchy → **OOP**
- Behavior & Data
- Positives:
  - Simple to Reason About
  - General Experience
  - Single Package
  - Supported by Default
  - Object Identity
- Negatives:
  - Multiple Inheritance
  - Size of Hierarchy
  - Hierarchy Depth
  - Growing Complexity
  - Monolith Anti-Pattern
  - Data Duplication
  - Parallelism & Serialization
  - Memory Layout

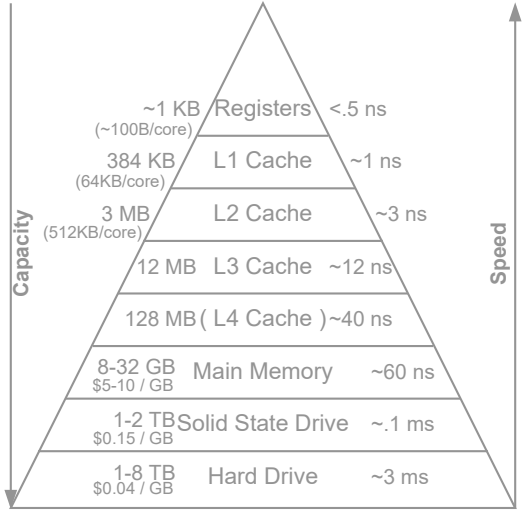



```
class Broom : Item
{ Broom() { mCost = 12; } }
class Broom : EnemyItem
{ Broom() { mHealth = 3; mCost = 12; } }
```

Consider Memory

- Hierarchy of Memory
- Why? → Memory Architecture [2]
- Caching & Locality
- Memory Gap





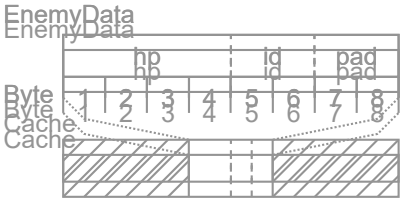
Source: CS 131: Fundamentals of Computer Systems

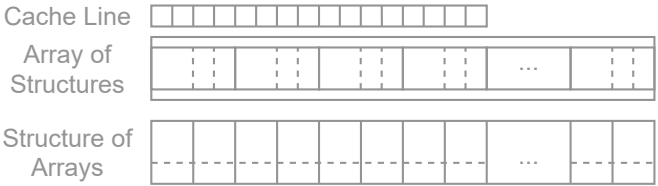
Data Oriented Design

- "**Getting** the Data is **the Problem**"
- DOD = Focus on the Data
- Data Locality & Caching
- AoS vs SoA
- Data Use Pattern
- Parallel Processing

```
struct EnemyData
{ float hp; short id; byte pad[2]; }
```

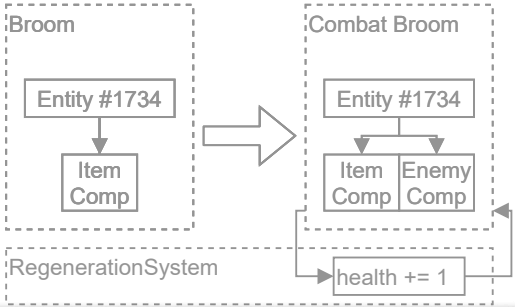
```
struct EnemyDatasAoS
{ EnemyData[] d; }
struct EnemyDataSoA
{ float[] hps; float[] ids; }
```





Hierarchy Through Composition

- Entity-Component-System
- **Entities** = “Instances”
- **Components** = Just Data
- **Systems** = Behavior & Logic
- **Blueprints** vs Classes
- Positives:
  - Implicit Decoupling
  - Parallelism & Serialization
  - Flat Hierarchy
  - Prevents Bloat
  - Cache Friendly
- Negatives:
  - Unusual Paradigm
  - New Anti-Patterns
  - Requires Implementation
  - Potential Overhead

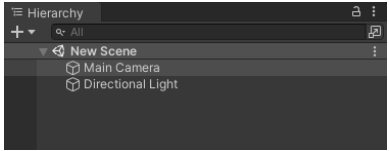


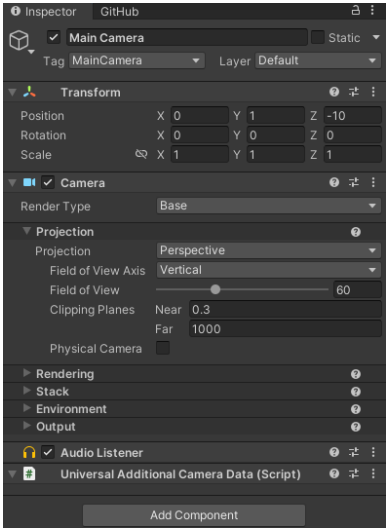

```
struct Entity
{ int id; }
struct ItemComponent struct EnemyComponent
{ int cost; }        { int health; }
class RegenerationSystem
{ (Entity e) =>
    { e<EnemyComponent>().health += 1; }
}
```

Hierarchy in Unity

GameObject Hierarchy

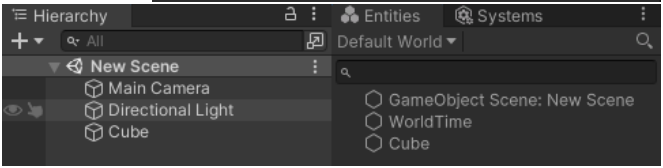
- "The Default"
- Hybrid **OOP** with **Components**
  - **GameObject** = Empty Container
  - **Component** as Base
  - **Behavior** can be Enabled
  - **MonoBehavior** adds Functionality
  - Not so Empty → **Transform**
- Editor Integration





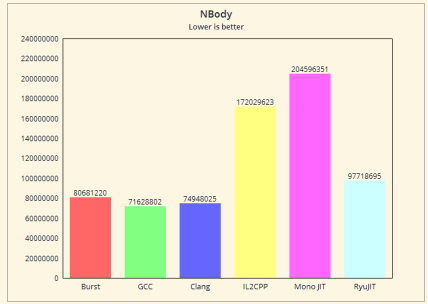
Entities Hierarchy

- "Experimental"
- Pure **Entity-Component-System**
  - **Entity** = Identifier
  - **Component** = Pure Data
  - **System** = Logic & Behavior
  - **World** = Entity Groups
- **Creation** vs **Conversion** Workflow
- Limited Editor Support
- → **DOTS**



Data-Oriented Technology Stack

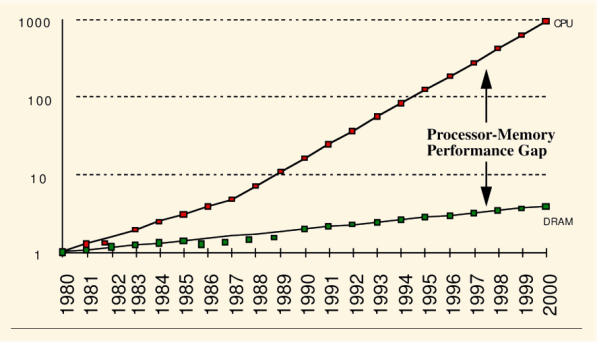
- Target → Performance
- Multi-Core & Acceleration
- Suit of Tools:
  - Entities
  - Collections
  - Mathematics
  - Jobs
  - Burst
  - And Others



Source: BurstBenchmarks GitHub

Additional Resources

- [Article] Robert Nystrom: Data Locality
- [Article] Robert Nystrom: Component
- [Book] Richard Fabian: Data-Oriented Design
- [YouTube] Luna Meier: Options for Entity Interaction



Source: A case for intelligent RAM


Thanks For
Your Attention!
Vampire: The Masquerade

References I

Logo: Adapted from work by Pablo Carlos Budassi and NASA

[1] Epic Games. ***Unreal Engine 4 Documentation.*** https://docs.unrealengine.com/.

[2] Jason Gregory. ***Game Engine Architecture, Second Edition.*** 3rd. USA: A. K. Peters, Ltd., CRC Press, 2018. ISBN: 1351974288.

[3] Juan Linietsky and Ariel Manzur. ***Godot Docs.*** https://docs.godotengine.org/en/stable/index.html.

[4] R. Nystrom. ***Game Programming Patterns.*** UK: Genever Benning, 2014. ISBN: 0990582906.

[5] Unity Technologies. ***Unity User Manual.*** https://docs.unity3d.com/Manual/index.html.