Lecture 03

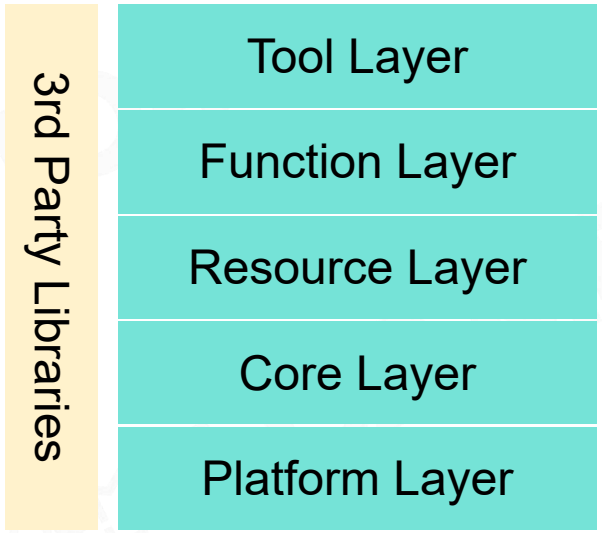
How to Build a Game World

Modern Game Engine - Theory and Practice

WANG XI

GAMES 104

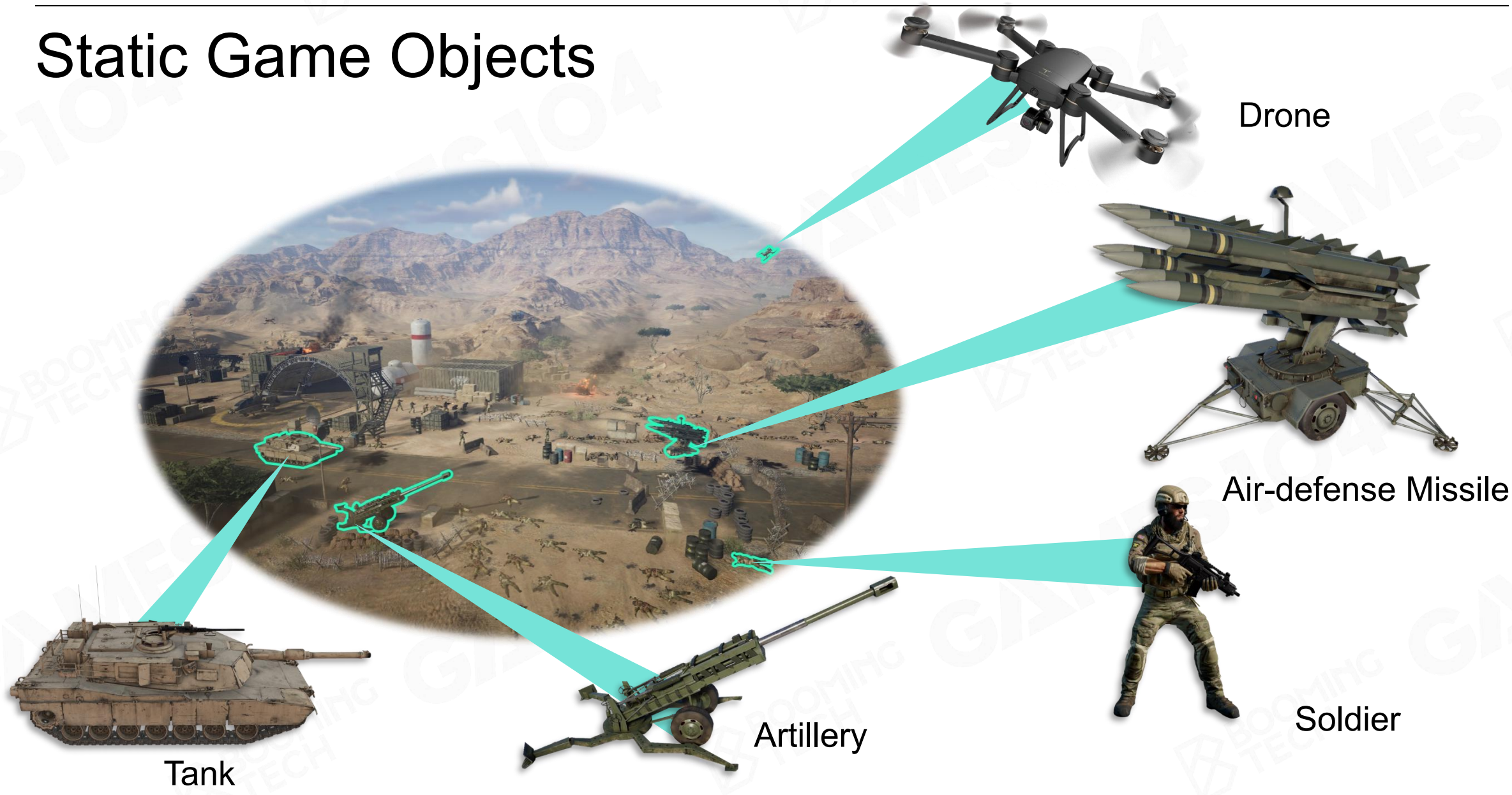
2022



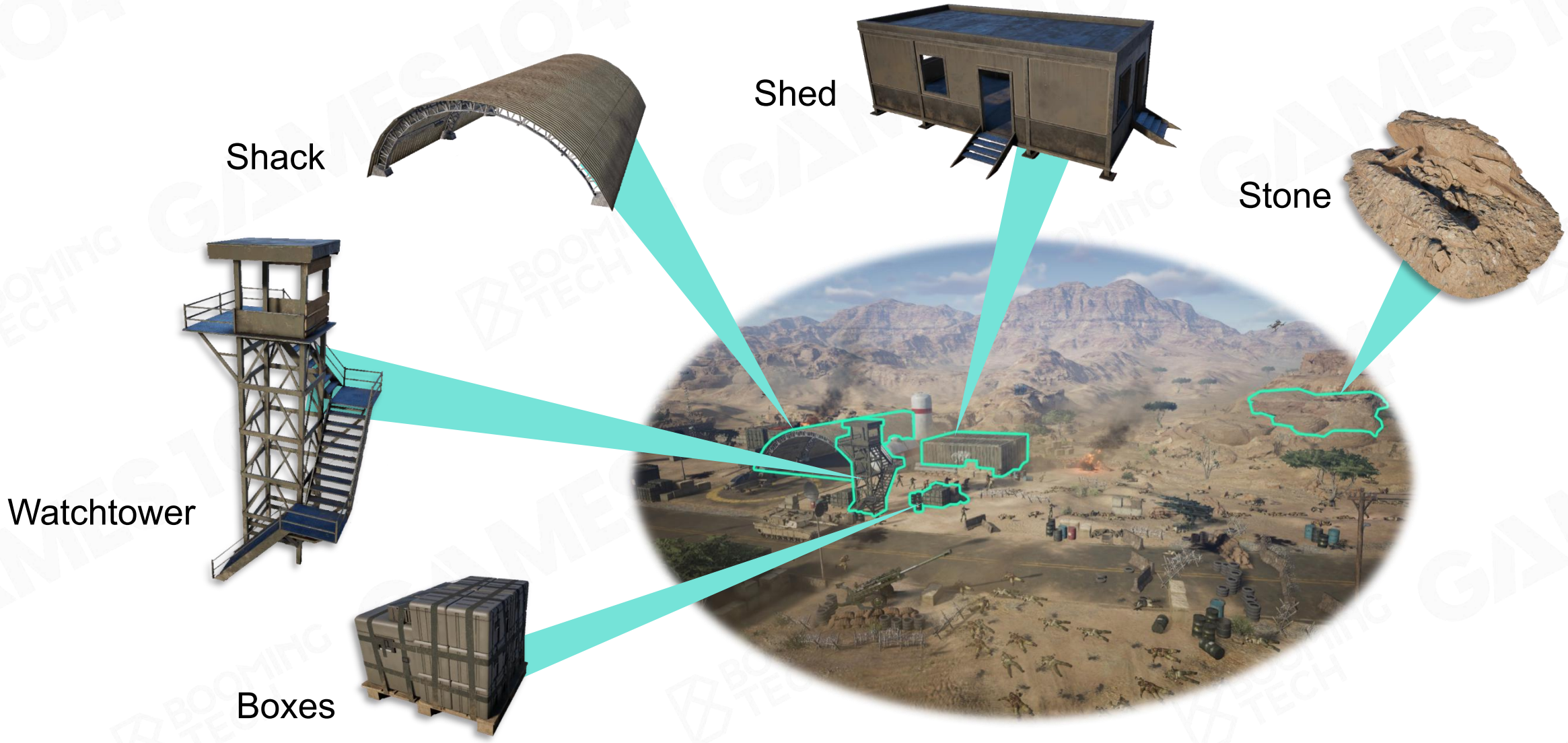
- What does a game world consist of?
- How should we describe these things?
- How are these things organized?

How to build a game world?

Static Game Objects

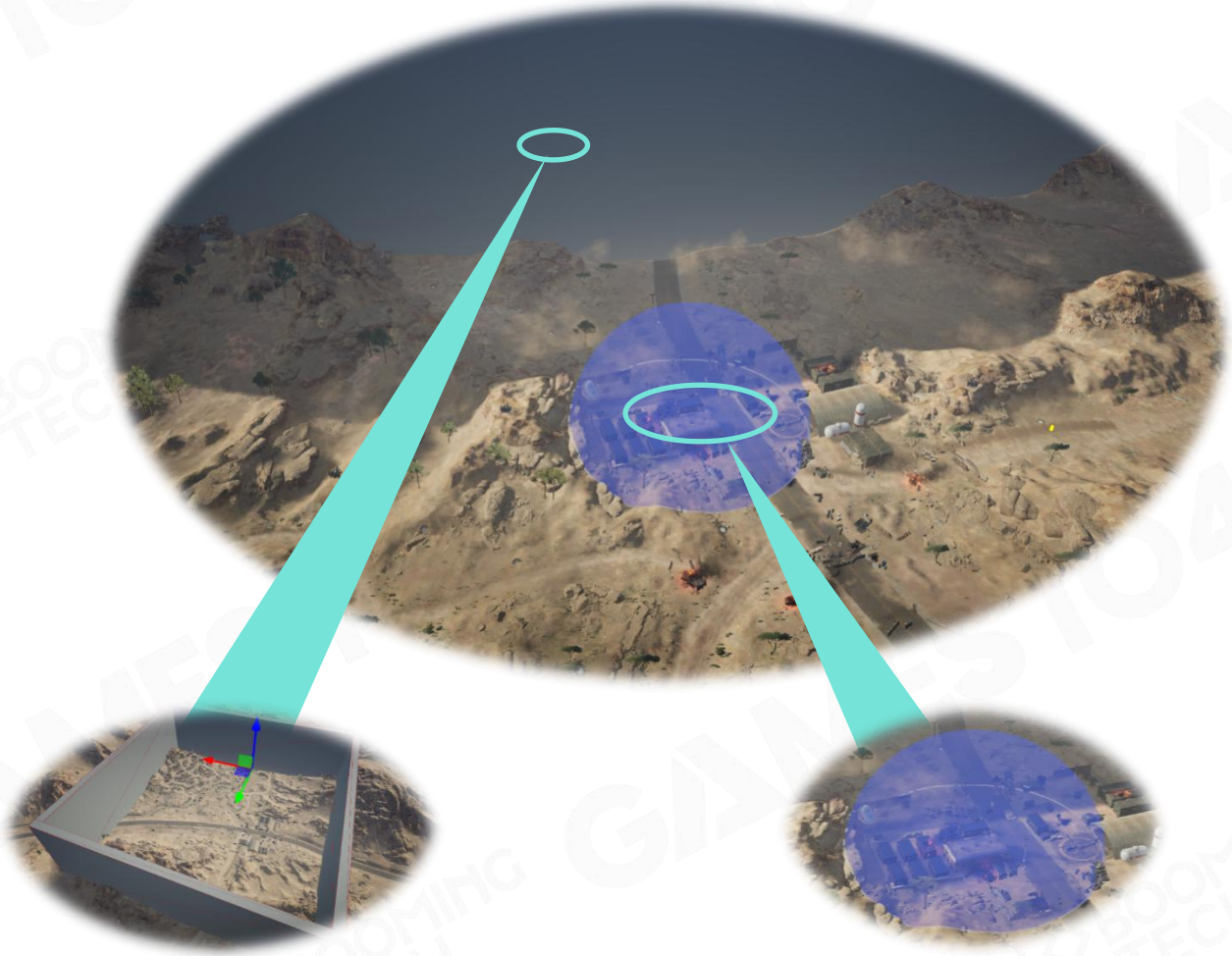


Static Game Objects



Environments

Other Game Objects



Air wall

Trigger Area

```
function ClientRulerBase:tick(delta_time)

    local os_utility = g_context.m_os_utility;
    local current_time_stamp = os_utility:getMil

    local current_level = g_context.m_level_mana
    local game_scene = current_level.m_scene;
    local scene_loading_status = game_scene:getS

    self:tickKickAFK(delta_time);
    self:tickBoss(delta_time);

    if self.m_game_status == ClientGameStatus._l
        local ruler_type_name = self.m_definitic
```

Ruler

Navigation mesh

Everything is a Game Object

Game Object (GO)

How to Describe a Game Object?

I want a drone!

How Do We Describe a Drone in Reality?

- Shape (property)
- Position (property)
- Move (behavior)
- Capacity of battery (property)
- Etc.

Game Object

Name	Drone
Property	position
	health
	battery
Behavior	move
	scout


```
class Drone
{
public:
    /* Properties */
    Vector3 position;
    float health;
    float fuel;
    ...
    /* Bebavior */
    void move();
    void scout();
    ...
};
```

Drone vs. Armed Drone

```
class Drone
{
public:
    /* Properties */
    Vector3 position;
    float health;
    float fuel;
    ...
    /* Bebavior */
    void move();
    void scout();
    ...
};
```

Name	Drone	ArmedDrone
Property	position	position
	health	health
	battery	battery
		ammo
Behavior	move	move
	scout	scout
		fire

```
class ArmedDrone
{
public:
    /* Properties */
    Vector3 position;
    float health;
    float fuel;
    float ammo;
    ...
    /* Bebavior */
    void move();
    void scout();
    void fire();
    ...
}
```

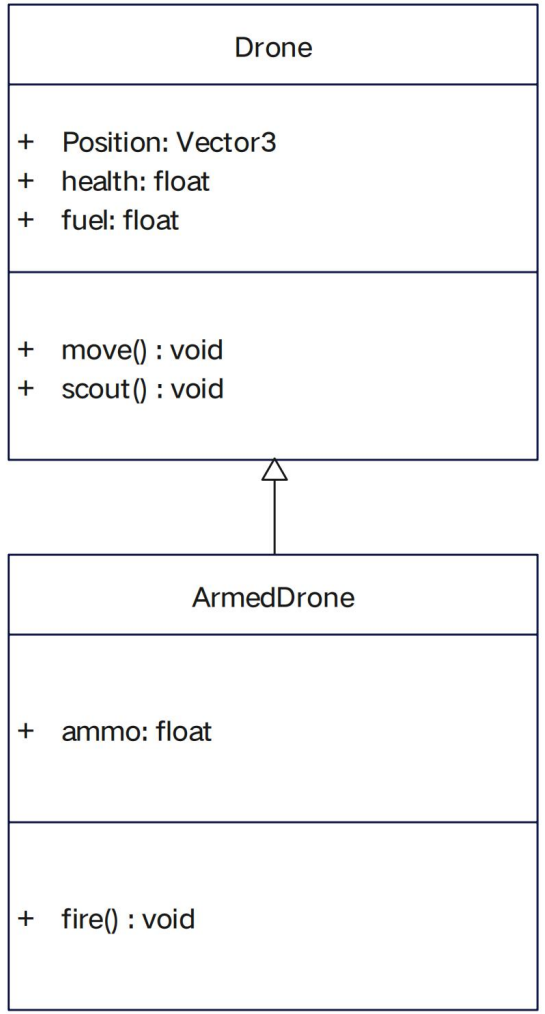
Game Object

- Inheritance

```
class Drone
{
public:
    /* Properties */
    Vector3 position;
    float health;
    float fuel;
    ...
    /* Bebavior */
    void move();
    void scout();
    ...
};
```

```
class ArmedDrone:
    public Drone
{
public:
    float ammo;
    void fire();

};
```

Code



UML Class Diagram

No Perfect Classification in the Game World!

Component Base

- Component Composition in the Real World

Excavator

Loader

Road Roller

Excavator

Bulldozer

Component Base

- Component Composition in the Real World

AR-15

M4

HK416

HK433

Components of a Drone

Component

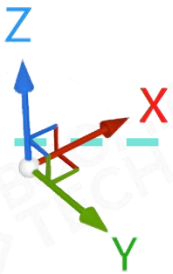
- Code example

Base class of component

```
class ComponentBase
{
    virtual void tick() = 0;
    ...
};
```

```
class GameObject
{
    vector<ComponentBase*> components;
    virtual void tick();
    ...
};
```

```
class TransformComponent:
    public ComponentBase
{
    Vector3 position;
    ...
    void tick();
};
```




```
class ModelComponent:
    public ComponentBase
{
    Mesh mesh;
    ...
    void tick();
};
```

```
class MotorComponent:
    public ComponentBase
{
    float battery;
    void tick();
    void move();
    ...
};
```

```
class AIComponent:
    public ComponentBase
{
    void tick();
    void scout();
    ...
};
```

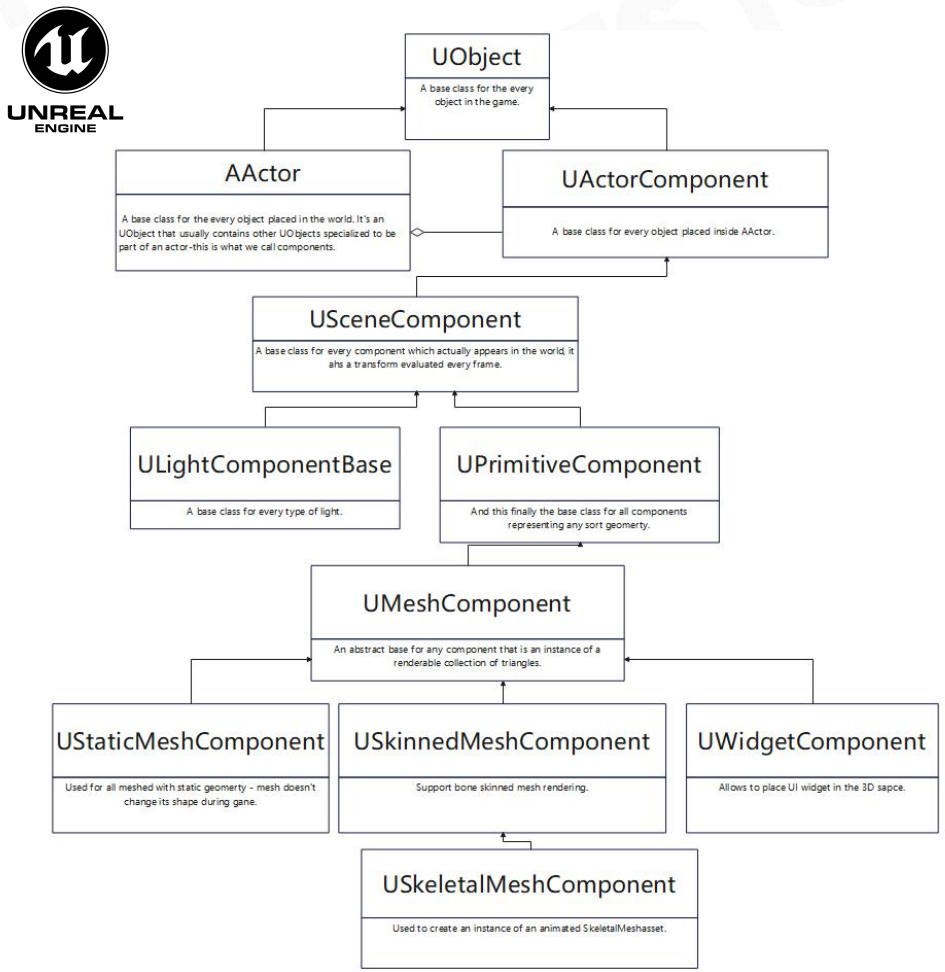
Animation
Physics
...

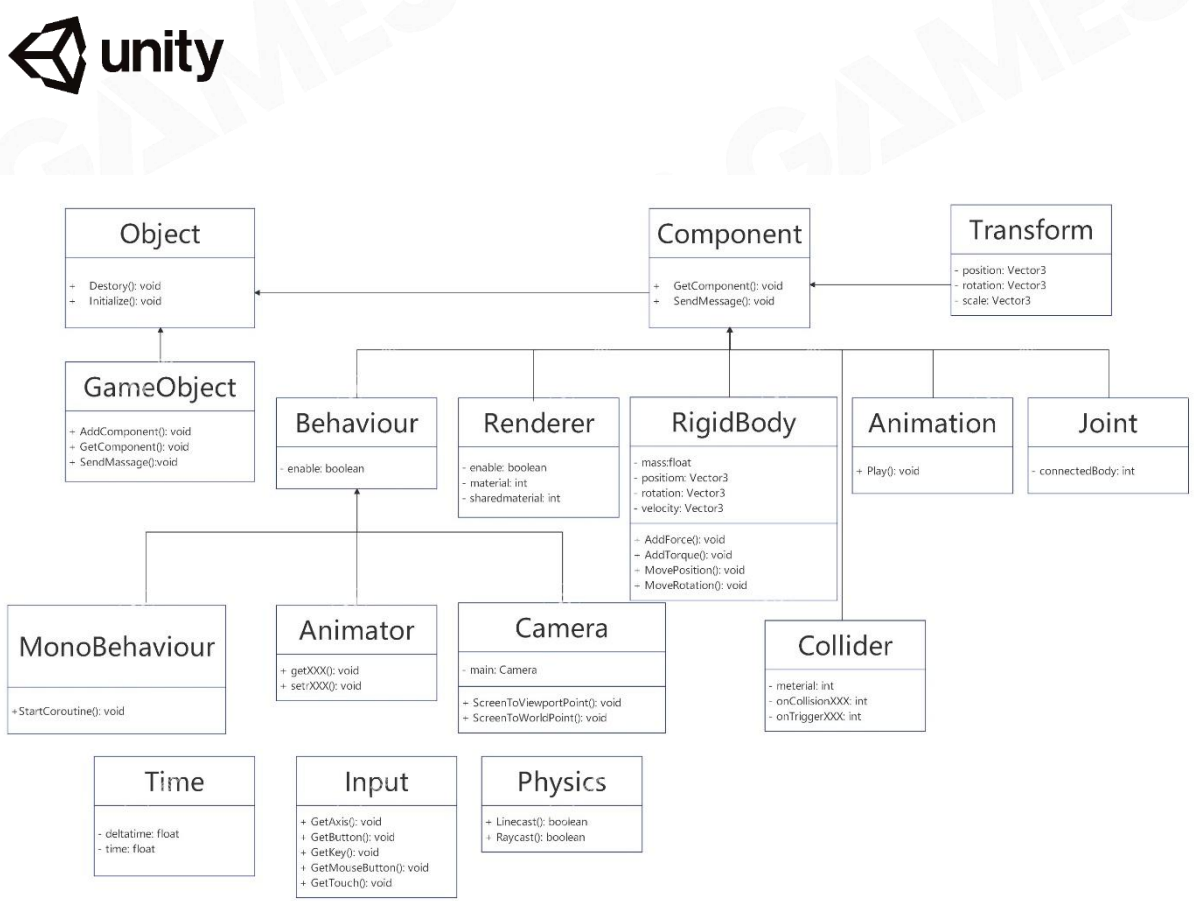
Component

- Drone vs. Armed Drone

Components		Components
Transform	=	Transform
Model	=	Model
Animation	=	Animation
Motor	=	Motor
Physics	=	Physics
…	=	…
AI	≠	AI
		Combat

Components in Commercial Engines





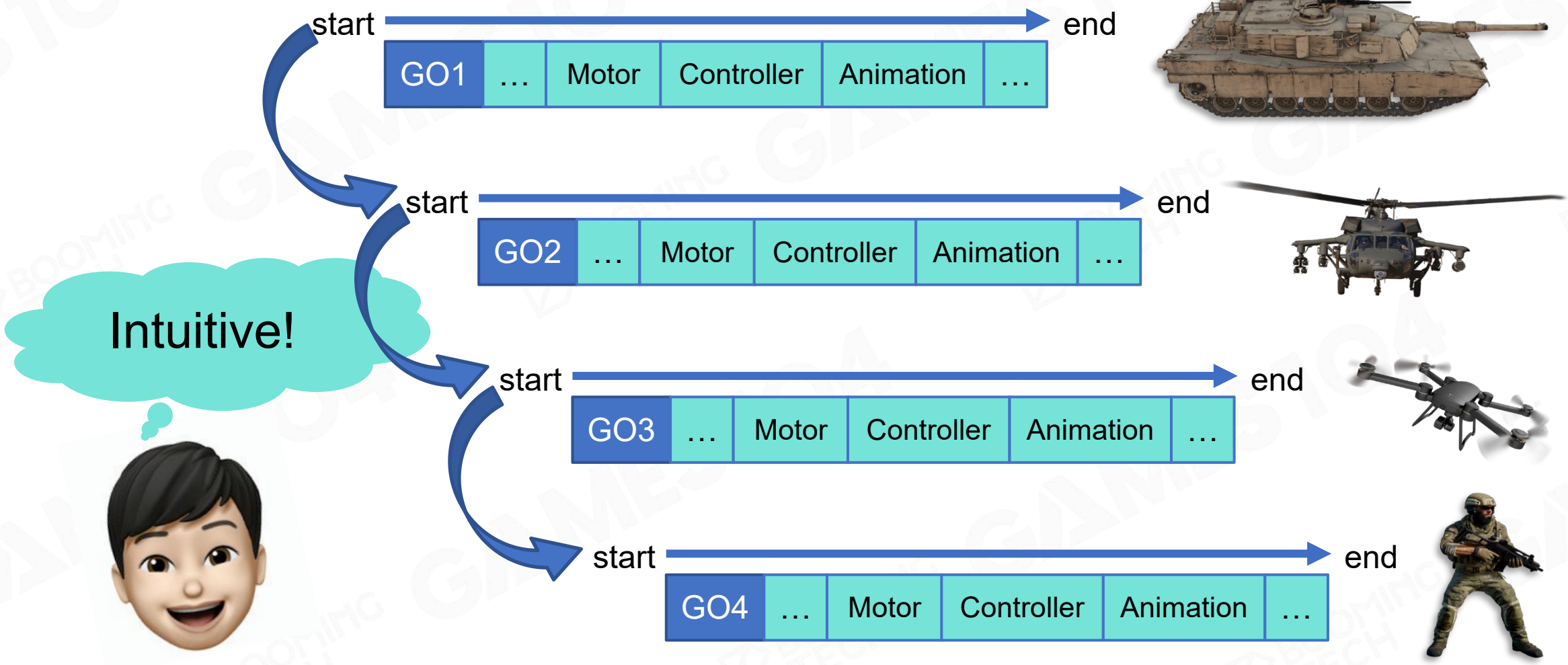
Takeaways

- Everything is a game object in the game world

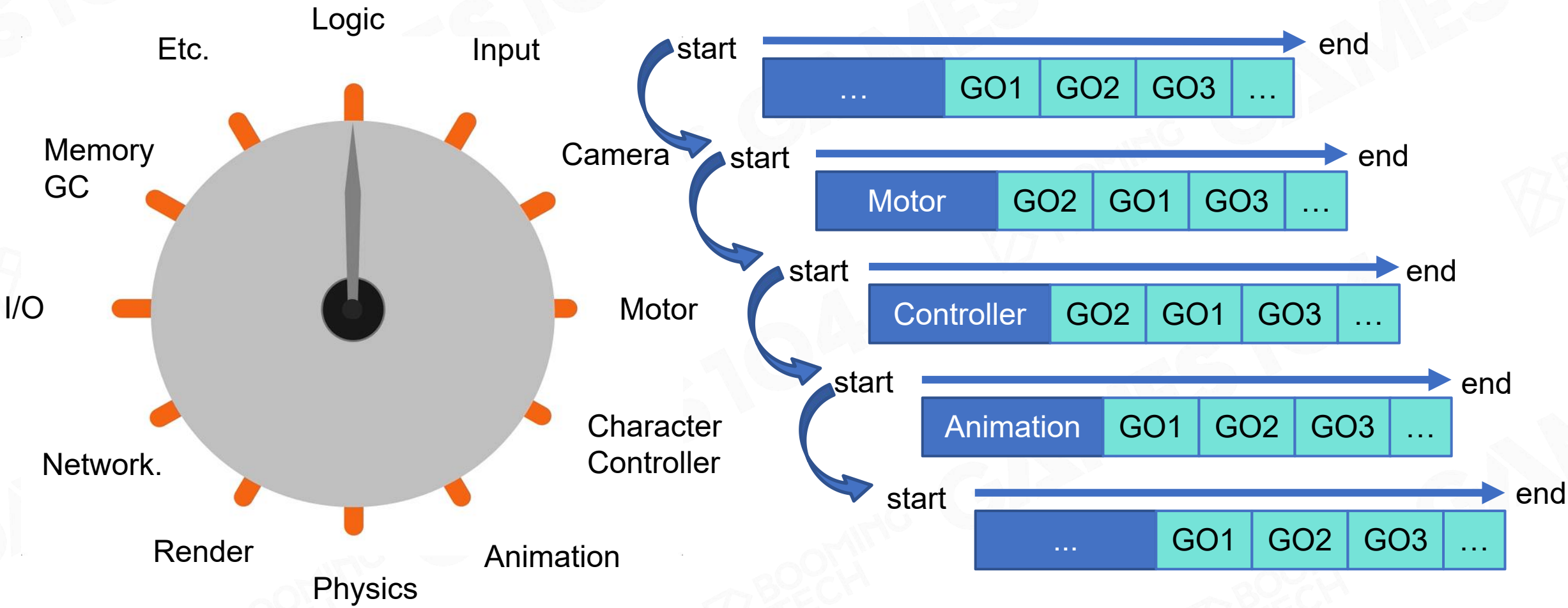
- Game object could be described in the component-based way

How to Make the World Alive?

Object-based Tick

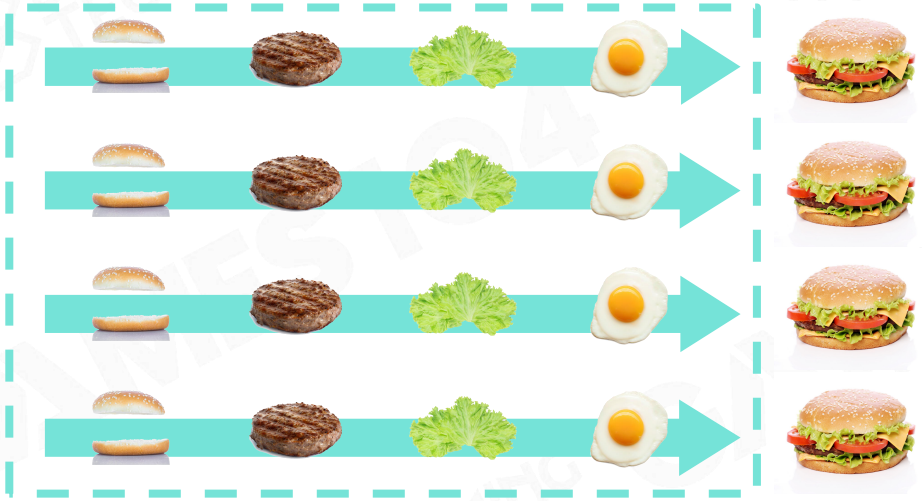


Component-based Tick

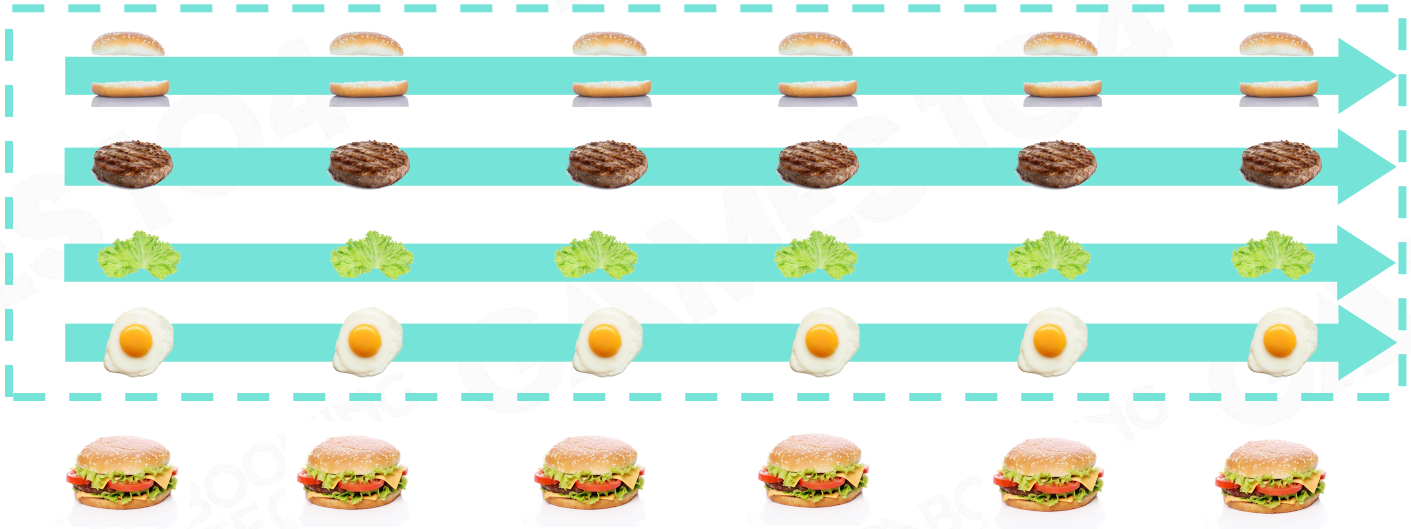


Object-based Tick vs. Component-based Tick

- Object-based tick
  - Simple and intuitive
  - Easy to debug



- Component-based tick
  - Parallelized processing
  - Reduced cache miss



How to Explode an Ammo in a Game?

Hardcode

Soldier

```
void Bomb:explode()
{
    ...
    switch(go_type)
    {
        case GoType.humen_type:
        {
            /* process soldier */
            ...
        }
        case GoType.drone_type:
        {
            /* process drone */
            ...
        }
        case GoType.tank_type:
        {
            /* process tank */
            ...
        }
        case GoType.stone_type:
        {
            /* process stone */
            ...
        }
        default:
        {
            break;
        }
    }
}
```

Helicopter

Tank

Stone

Events

- Message sending and handling
- Decoupling event sending and handling

Soldier

```
void Bomb:explode()
{
    ...
    sendExplodeEvent(go_id)
    ...
}
```

Helicopter

Tank

Stone

Events Mechanism in Commercial Engines

unity

```
using UnityEngine;

public class Example : MonoBehaviour
{
    void Start()
    {
        // Calls the function ApplyDamage with a value of 5
        // Every script attached to the game object
        // that has an ApplyDamage function will be called.
        gameObject.SendMessage("ApplyDamage", 5.0);
    }
}

public class Example2 : MonoBehaviour
{
    public void ApplyDamage(float damage)
    {
        print(damage);
    }
}
```

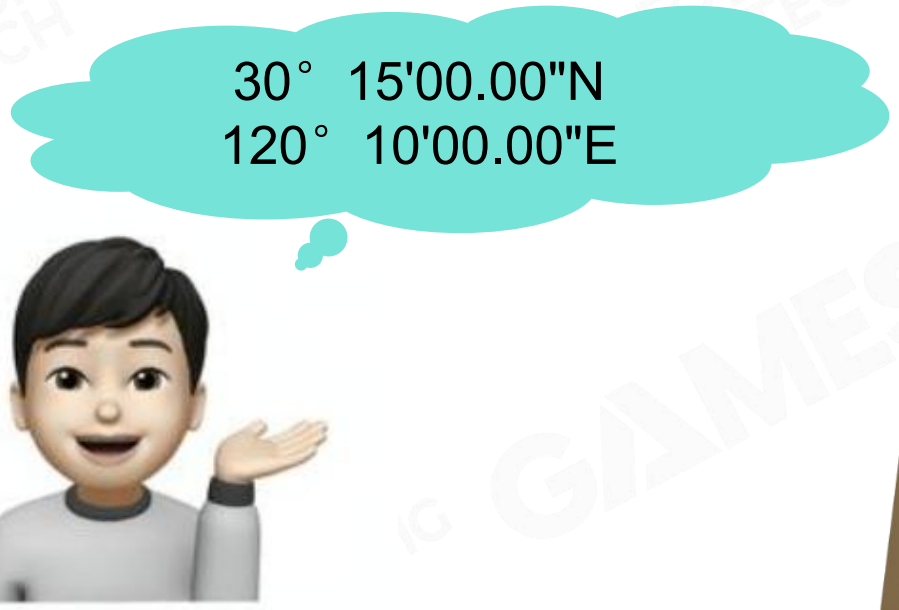
UNREAL ENGINE

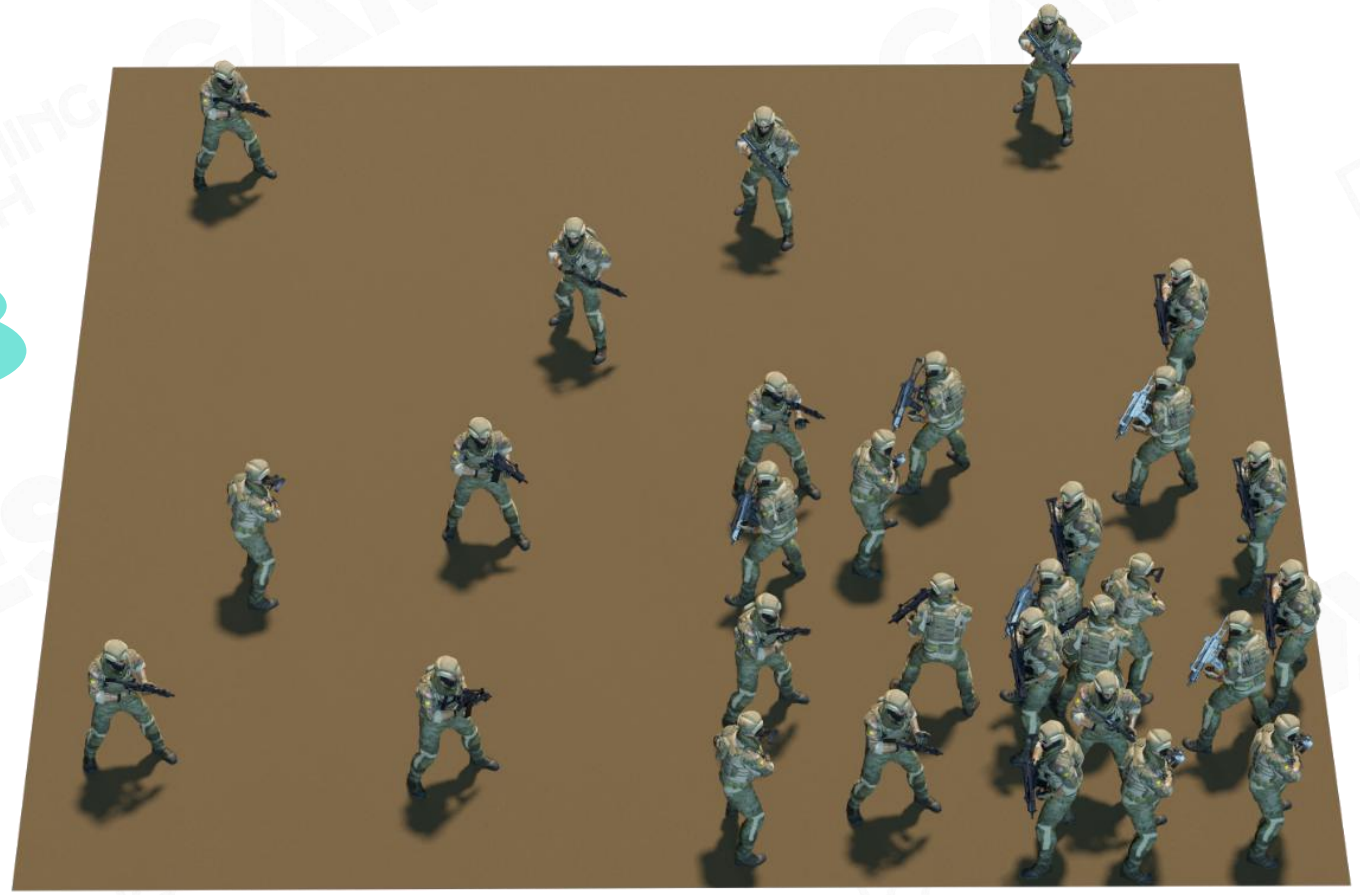
```
/** Event for when collections are created */
DECLARE_EVENT_OneParam( ICollectionManager, FCollectionCreatedEvent, const FCollectionNameType& );
virtual FCollectionCreatedEvent& OnCollectionCreated() = 0;

/** Event for when collections are destroyed */
DECLARE_EVENT_OneParam( ICollectionManager, FCollectionDestroyedEvent, const FCollectionNameType& );
virtual FCollectionDestroyedEvent& OnCollectionDestroyed() = 0;
  Tim Sweeney, 8年前 • Engine source (Main branch up to CL 2026164) …
/** Event for when assets are added to a collection */
DECLARE_EVENT_TwoParams( ICollectionManager, FAssetsAddedEvent, const FCollectionNameType&, const TArray<FName>& );
virtual FAssetsAddedEvent& OnAssetsAdded() = 0;

/** Event for when assets are removed from a collection */
DECLARE_EVENT_TwoParams( ICollectionManager, FAssetsRemovedEvent, const FCollectionNameType&, const TArray<FName>&
virtual FAssetsRemovedEvent& OnAssetsRemoved() = 0;

/** Event for when collections are renamed */
DECLARE_EVENT_TwoParams( ICollectionManager, FCollectionRenamedEvent, const FCollectionNameType&, const FCollectionN
virtual FCollectionRenamedEvent& OnCollectionRenamed() = 0;

/** Event for when collections are re-parented (params: Collection, OldParent, NewParent) */
DECLARE_EVENT_ThreeParams( ICollectionManager, FCollectionReparentedEvent, const FCollectionNameType&, const TOptior
virtual FCollectionReparentedEvent& OnCollectionReparented() = 0;

/** Event for when collections is updated, or otherwise changed and we can't tell exactly how (eg, after updating fr
DECLARE_EVENT_OneParam( ICollectionManager, FCollectionUpdatedEvent, const FCollectionNameType& );
virtual FCollectionUpdatedEvent& OnCollectionUpdated() = 0;

/** When a collection checkin happens, use this event to add additional text to the changelist description */
DECLARE_EVENT_TwoParams( ICollectionManager, FAddToCollectionCheckinDescriptionEvent, const FName& /*CollectionName*
virtual FAddToCollectionCheckinDescriptionEvent& OnAddToCollectionCheckinDescriptionEvent() = 0;
};
```

How to Manage Game Objects?

Scene Management

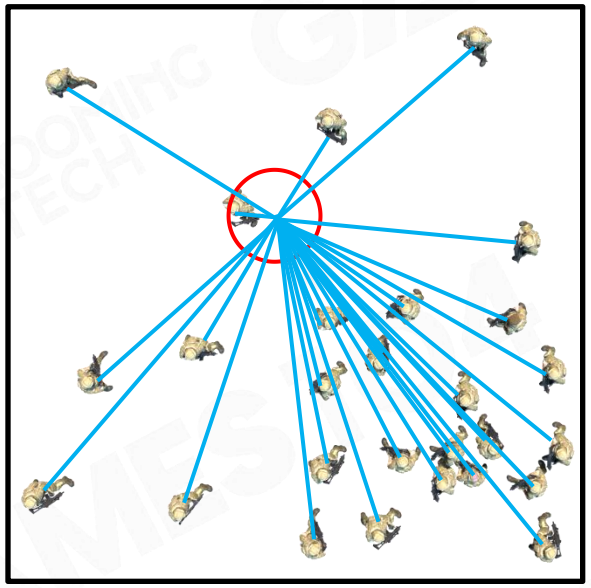
- Game objects are managed in a scene
- Game object query
  - By unique game object ID
  - By object position



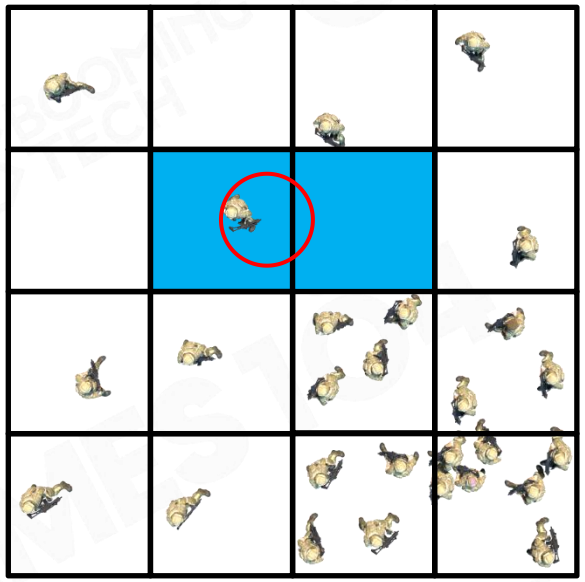


Scene Management

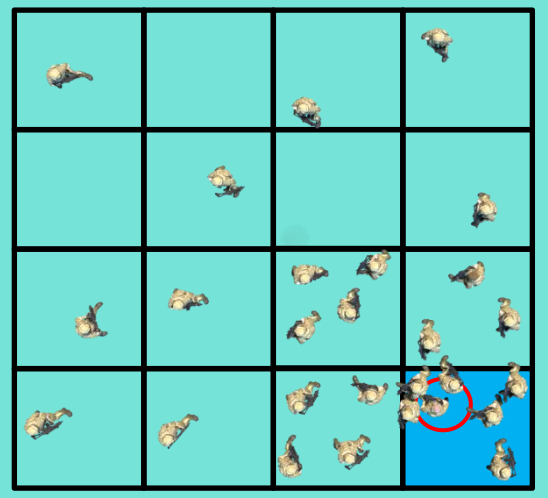
- Simple space segmentation



No division



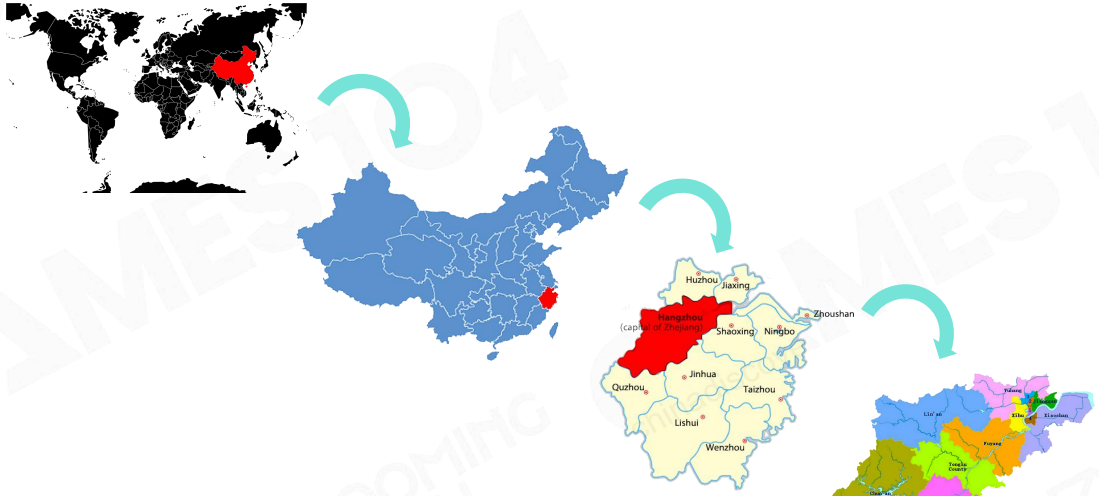
Divided by grid

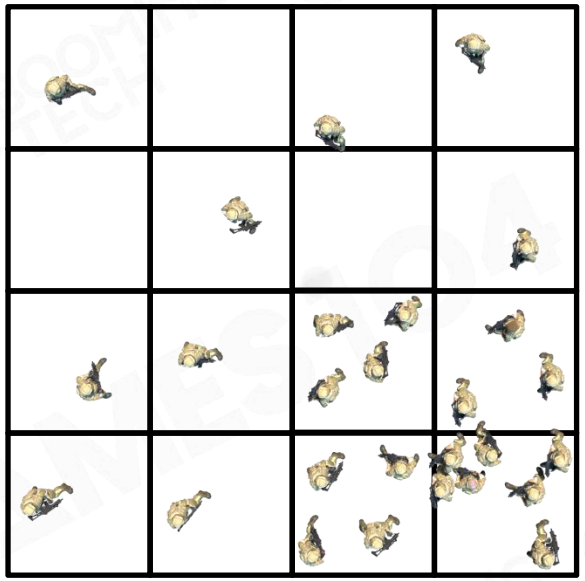


A needle in a haystack!

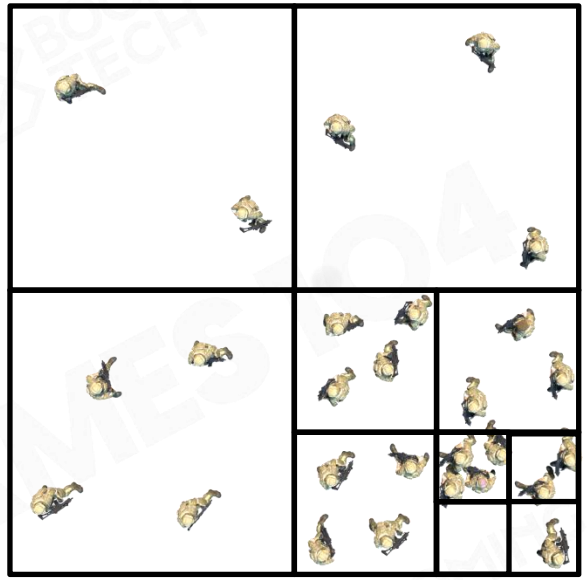
Scene Management

- Segmented space by object clusters
- Hierarchical segmentation





Divided by grid



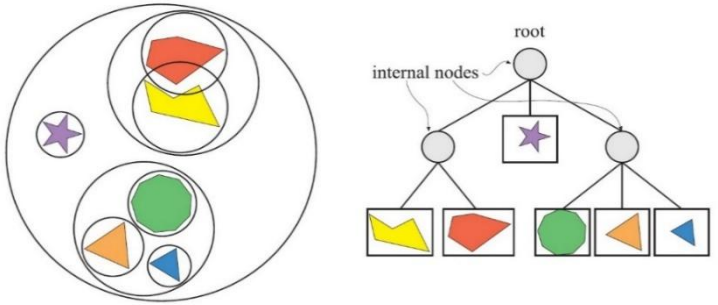
Quadtree

Scene Management

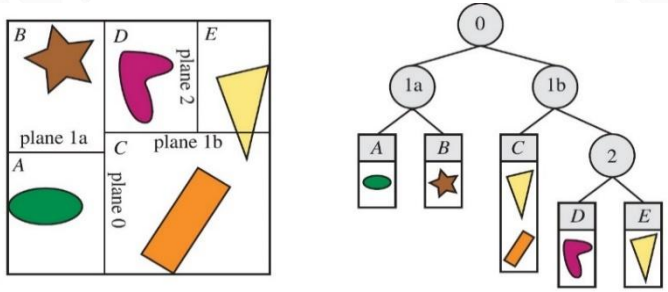
- Segmented space by object clusters
- Hierarchical segmentation

Scene Management

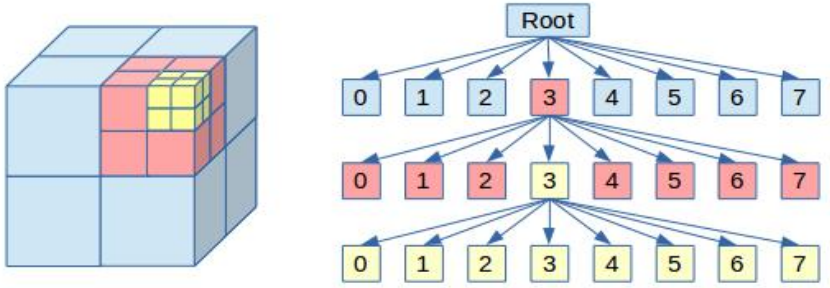
- Spatial Data Structures



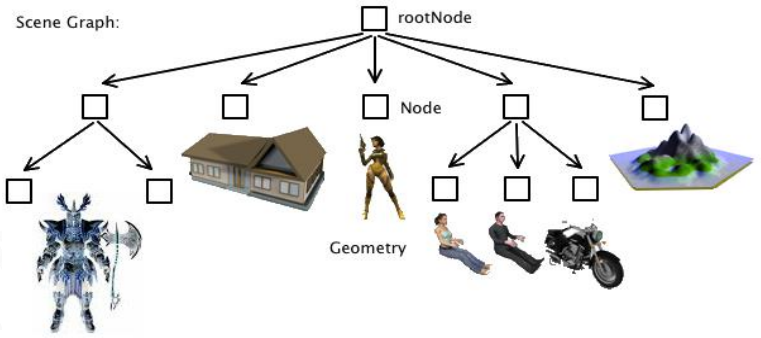
Bounding Volume Hierarchies (BVH)



Binary Space Partitioning(BSP)



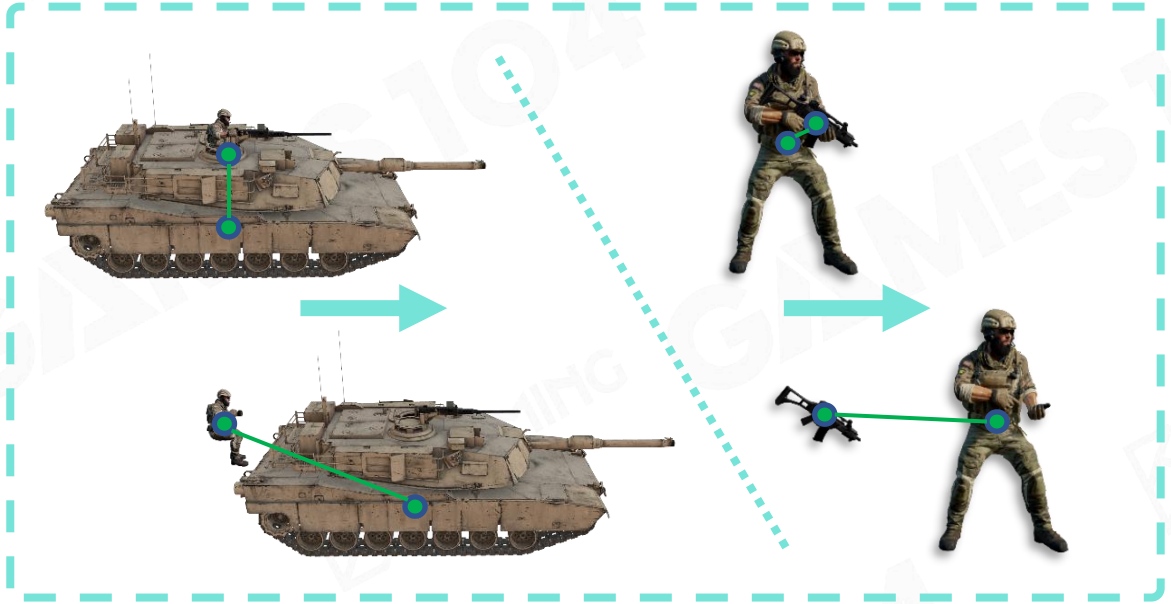
Octree



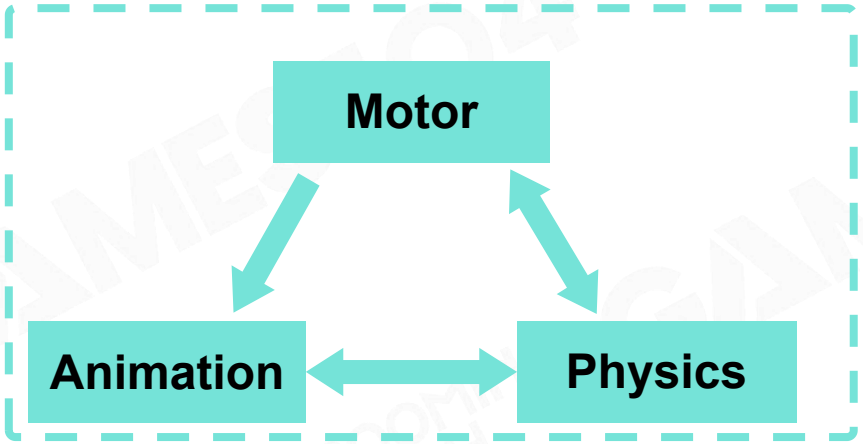
Scene Graph

Takeaways

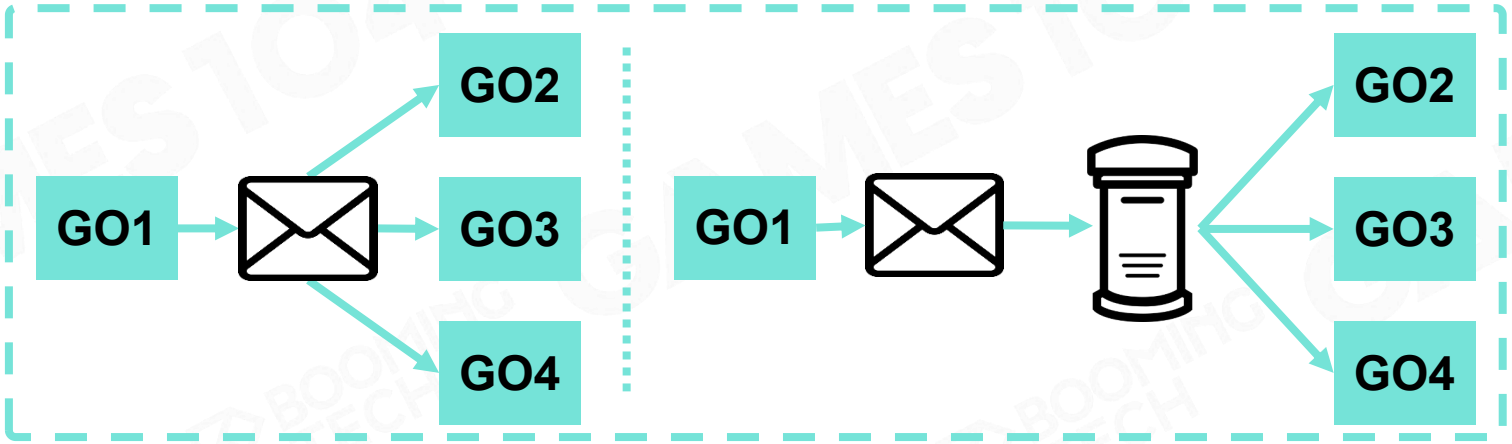
- Everything is an object
- Game object could be described in the component-based way
- States of game objects are updated in tick loops
- Game objects interact with each other via event mechanism
- Game objects are managed in a scene with efficient strategies



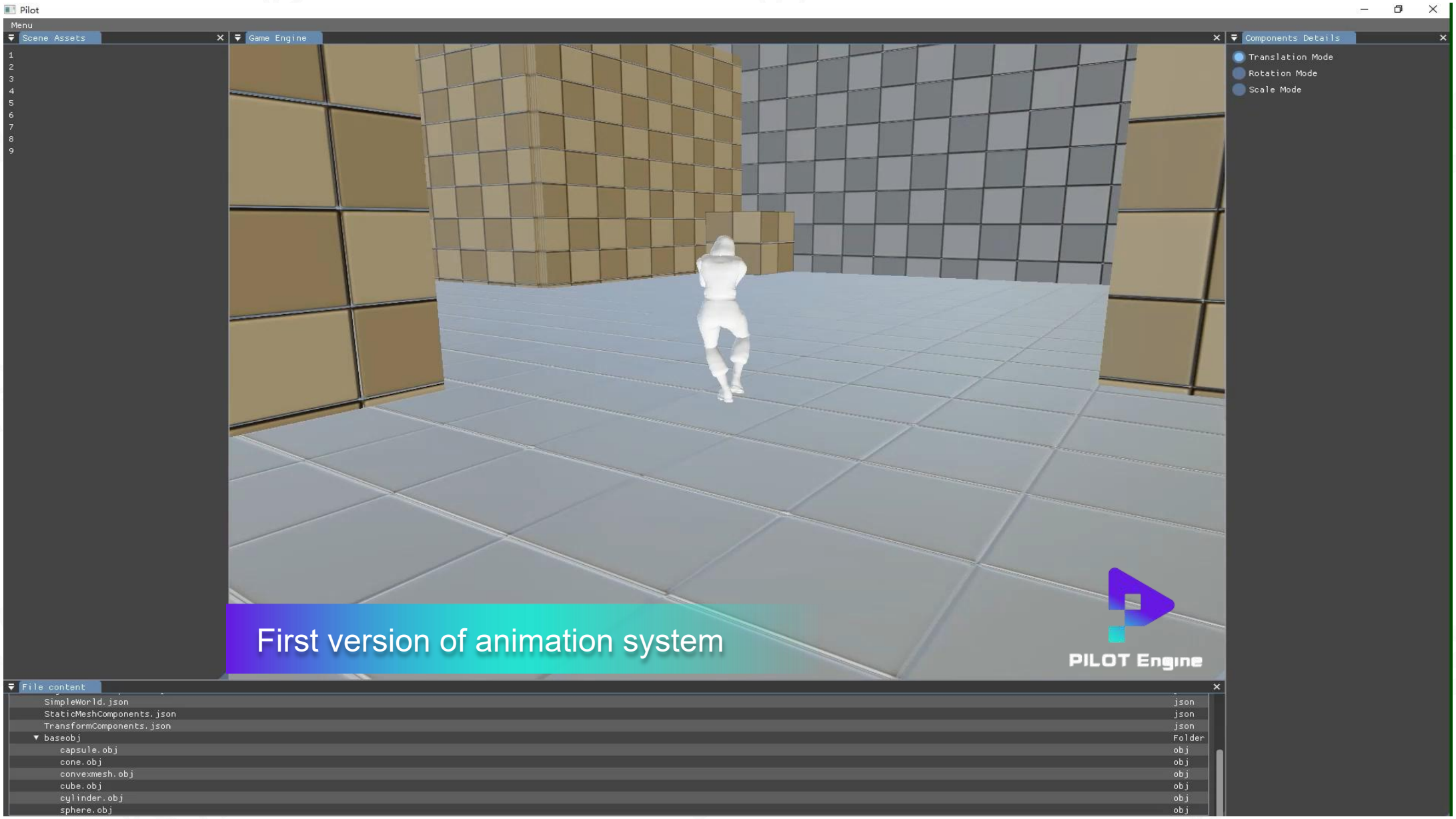
GO Bindings



Component Dependencies



Immediate Event Sending or not


Pilot
Menu
Scene Assets
Game Engine
Components Details
Translation Mode
Rotation Mode
Scale Mode
First version of animation system
PILOT Engine
File content
SimpleWorld.json
StaticMeshComponents.json
TransformComponents.json
baseobj
capsule.obj
cone.obj
convexmesh.obj
cube.obj
cylinder.obj
sphere.obj

Course Survey

Please scan here to let us know what you think

Lecture 03 Contributor

- 一将
- Hoya
- 喵小君
- 呆呆兽
- Olorin
- 靓仔
- 爵爷
- Jason
- 砚书
- BOOK
- MANDY
- 俗哥
- 金大壮
- Leon
- 梨叔
- Shine
- 邓导
- Judy
- QIUU
- C佬
- 阿乐
- 阿熊
- CC
- 大喷

Q&A

Enjoy ;)
Coding

Course Wechat

Please follow us for further information