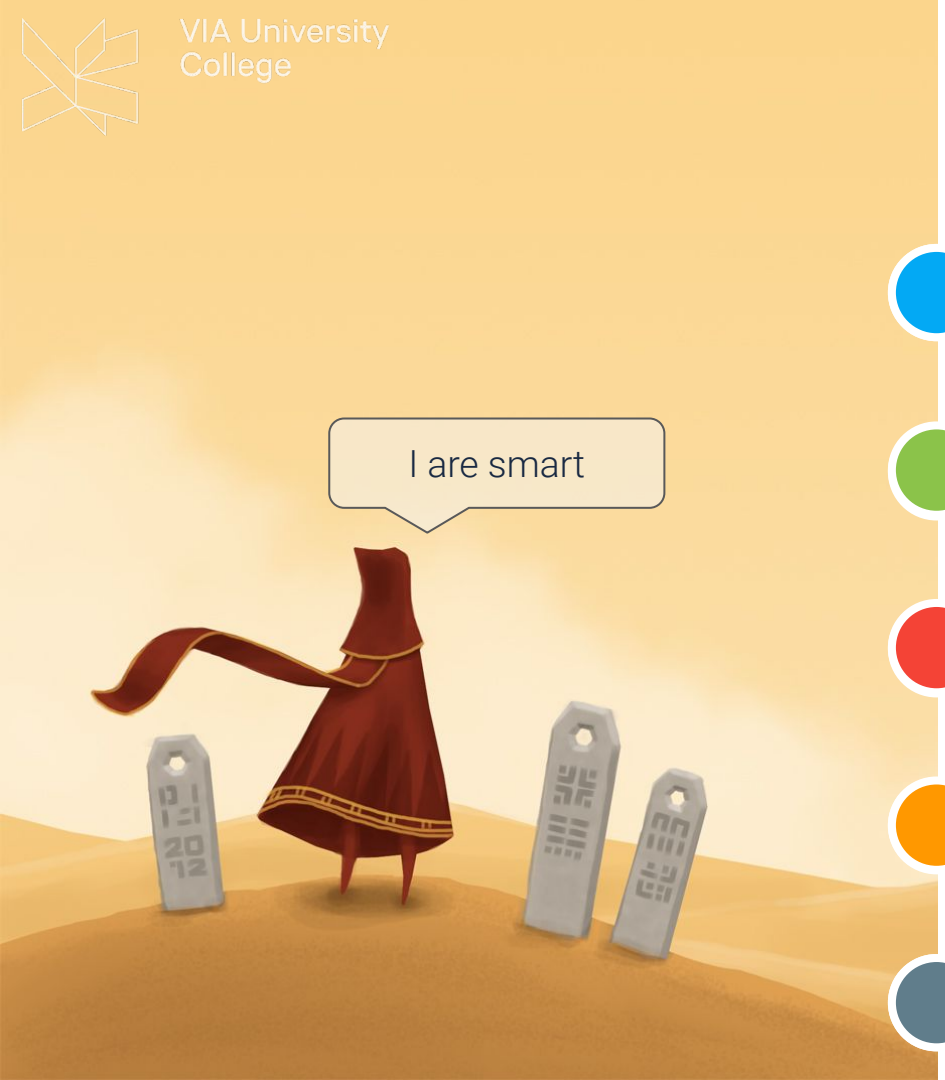
VIA University College

Game AI

Simulating Intelligence



Artificial Intelligence?

What is Game AI?

Simple Examples of Game AI

Example Project Demo

Coroutines

Modelling Behaviour over Time

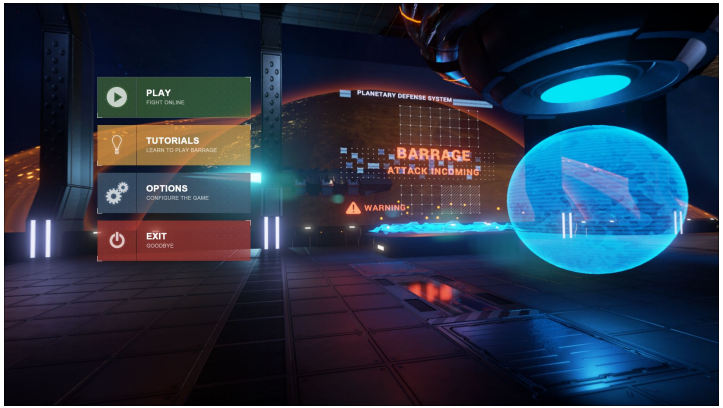
Pathfinding with NavMesh Agents

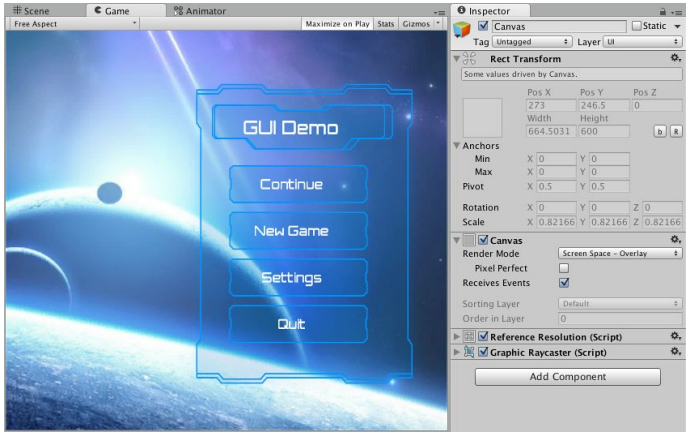
Making complex pathfinding easy

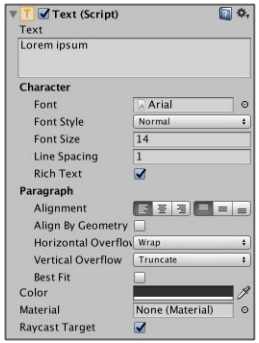
Exercises

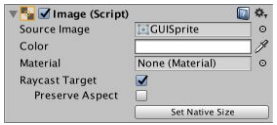
Create your own AI

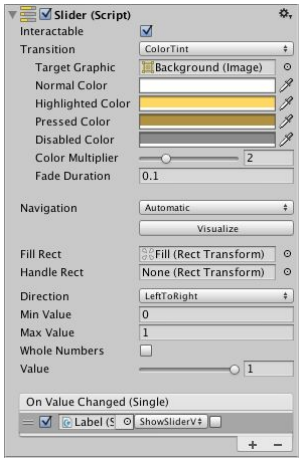
Last Week - What did we learn?

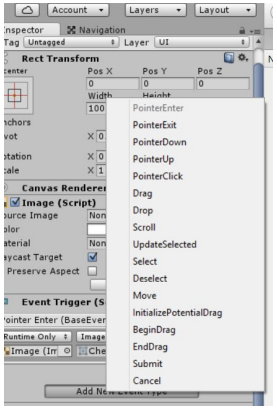


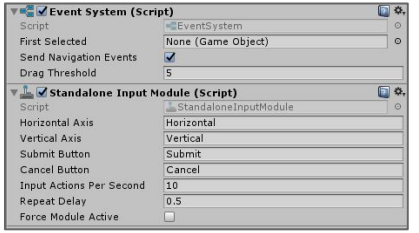


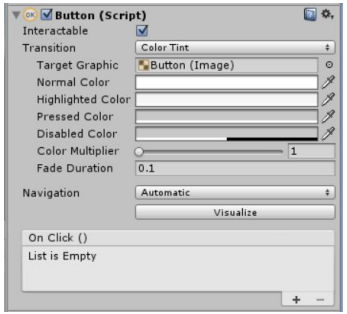












Artificial Intelligence?

“The intelligence demonstrated by a machine”

Reasoning

Knowledge representation

Planning

Learning

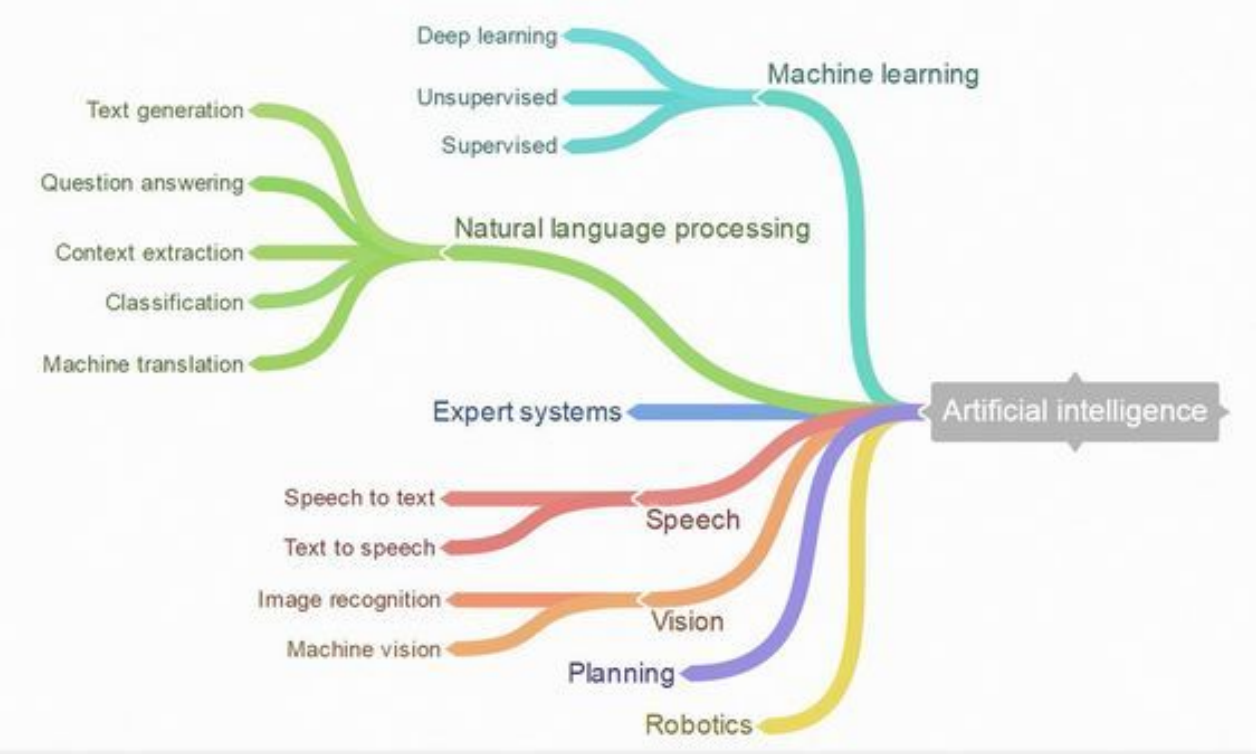
Natural language processing

Perception

Manipulation

Artificial Intelligence?

In order to solve the problem of AI, many fields have emerged



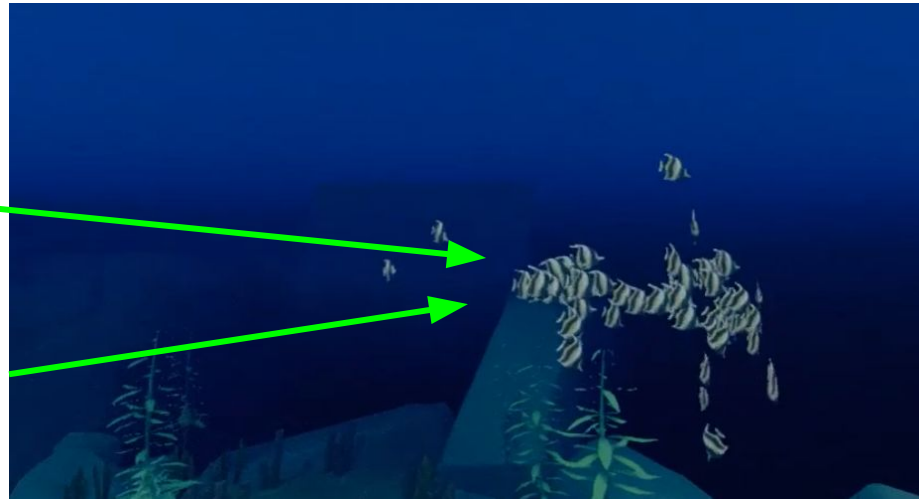
What is Game AI?

- Academic (“traditional”) AI != Game AI
- We want to **SIMULATE** intelligence
  - Create something that the player could be tricked into thinking is real
  - Making NPC characters that move around the environment, interact with the player

“exhibit traits of being digitally alive”

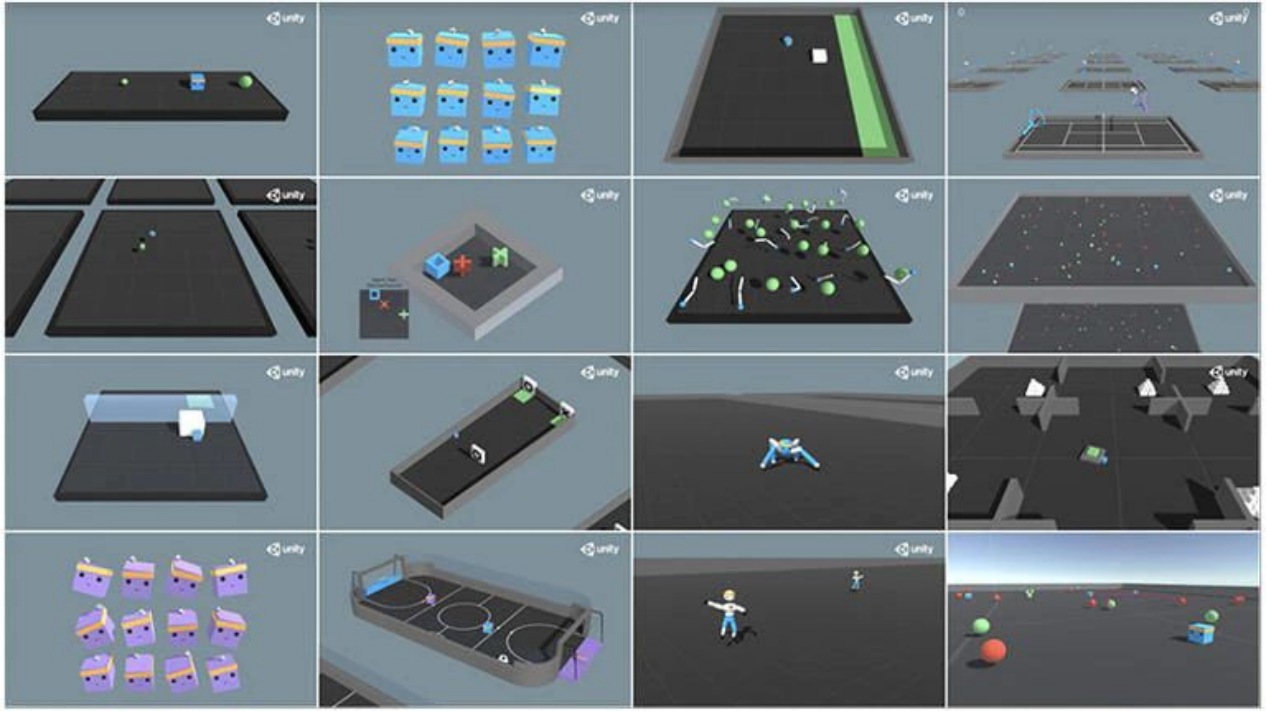
Intelligent?

Uses a relatively simply flocking algorithm (“boids”)



ML Agents

ML Agents are cool, but not really what we need for our games…



Simple Game AI Examples

Following enemy

Ranged enemy

Patrolling enemy

Lookout enemy

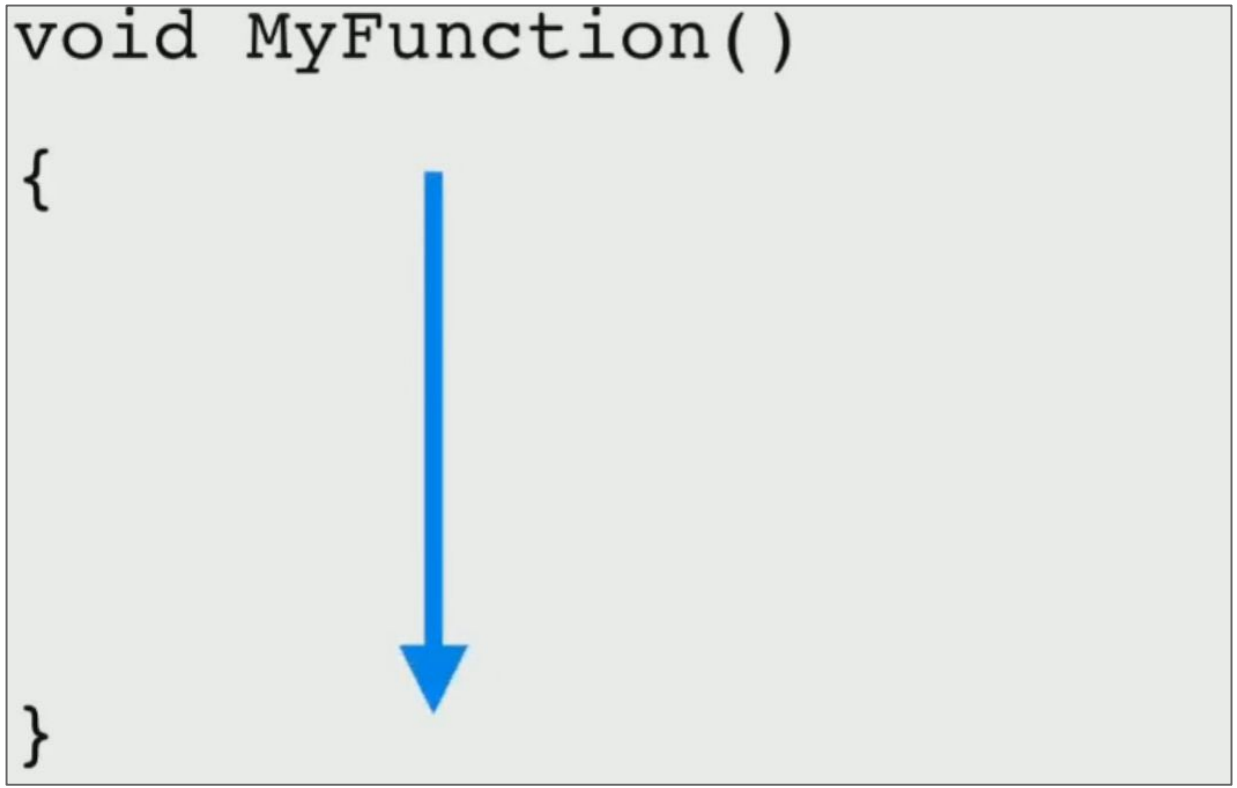
Pathfinding Enemy

Coroutines

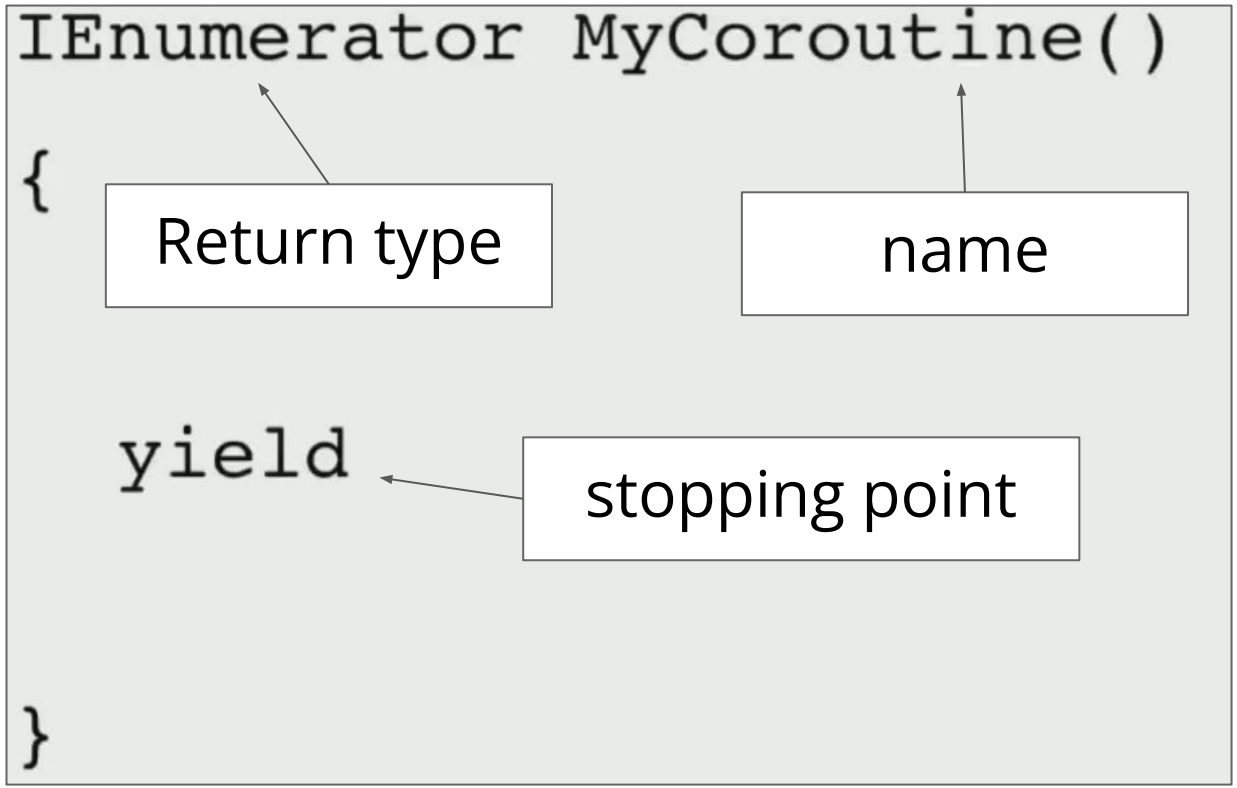
- When you call a function, it runs to completion before returning. This effectively means that **any action taking place in a function must happen within a single frame update**; a function call can't be used to contain a procedural animation or a sequence of events over time.
- **All Monobehaviour scripts must be executed on the main thread.**

- How do we then execute code that lasts longer than a single frame?
- How do we pause execution of code?

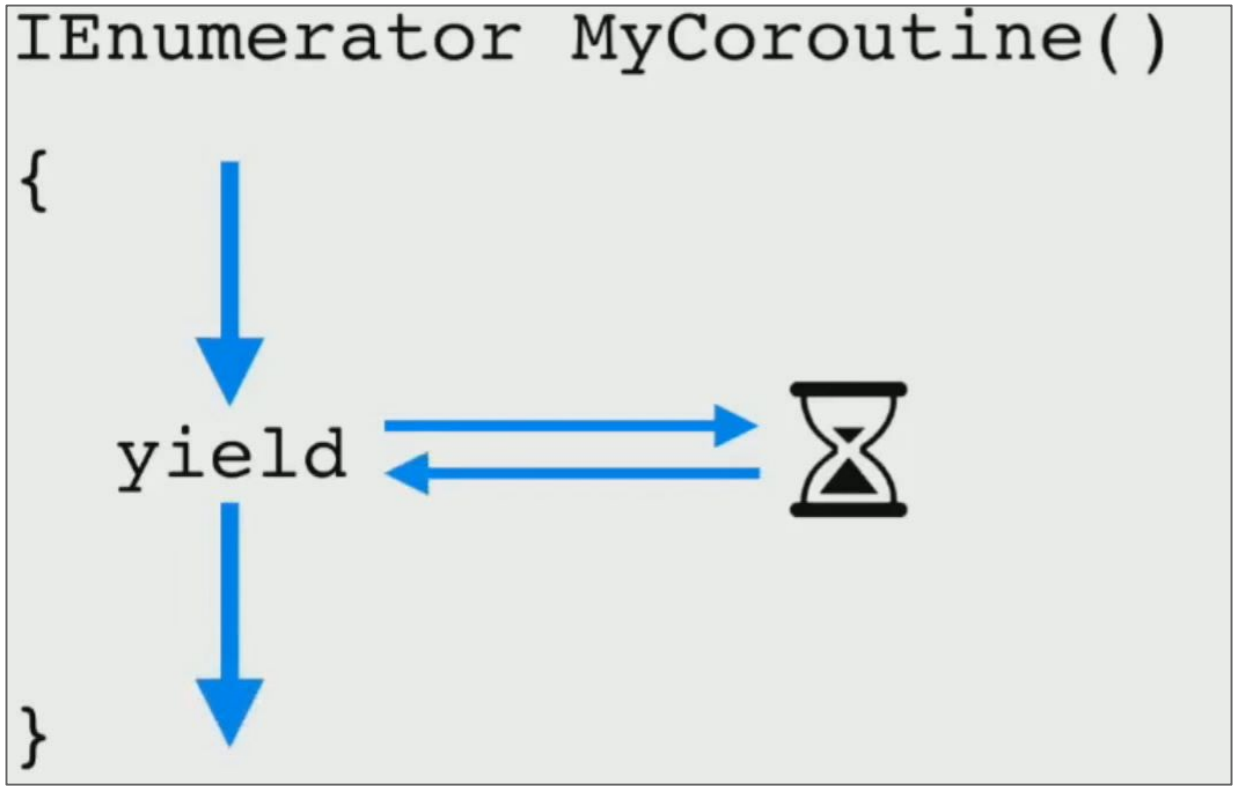
Coroutines



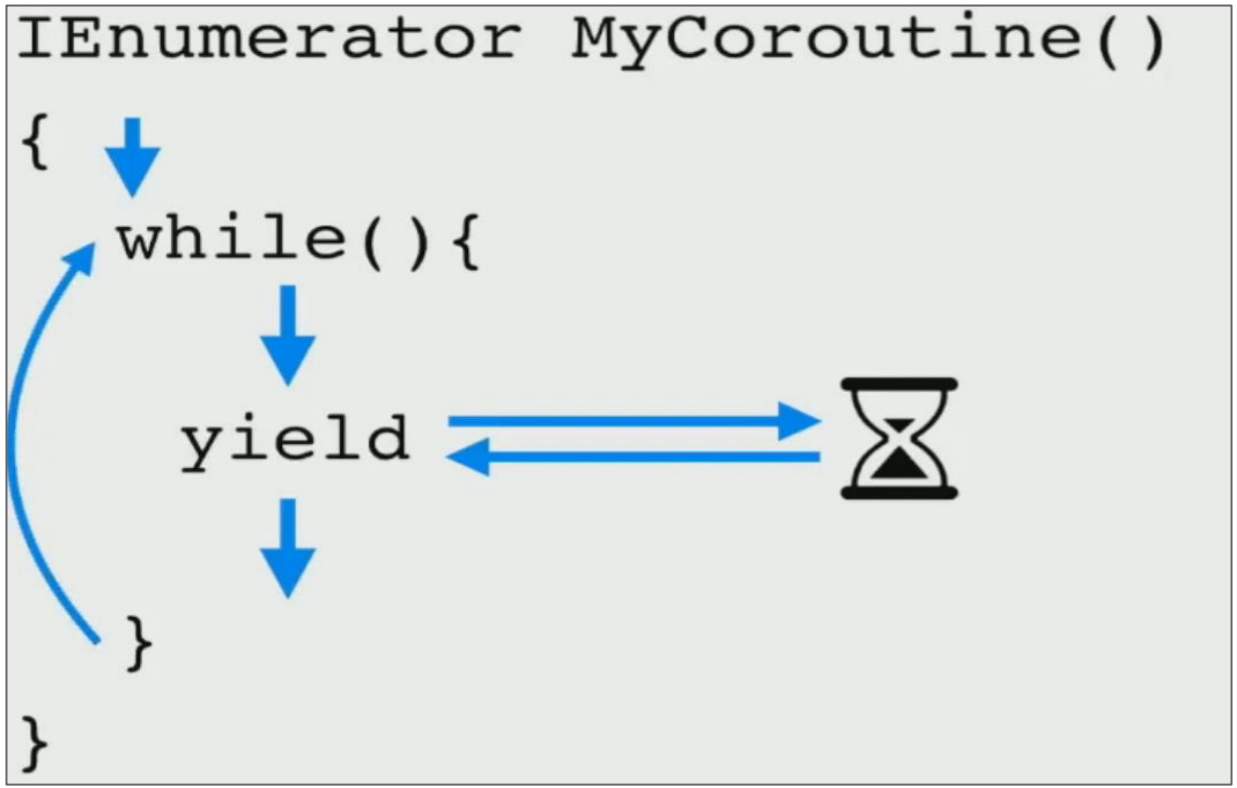
Coroutines


IEnumerator MyCoroutine()
{
Return type
name
yield
stopping point
}

Coroutines



Coroutines



Coroutines

Without coroutine

```
using UnityEngine;
using System.Collections;

public class example : MonoBehaviour {
    private int state = 0;
    void Update() {
        if (state == 0) {
            state = 1;
            return;
        }
        if (state == 1) {
            state = 2;
            return;
        }
    }
}
```

Do something here!

Do something here!

With coroutine

```
using UnityEngine;
using System.Collections;

public class example : MonoBehaviour {
    IEnumerator Example() {
        while (true) {
            yield return null;
            yield return null;
        }
    }
}
```

Do something here!

Do something here!

Coroutines

- A Coroutine is started with **StartCoroutine(name)**
- To repeat:
  - Coroutines are useful for pausing in the middle of your functions
  - Excellent when modelling behavior over several frames
  - Becomes very useful with while loops (and properties!)
- The yield return value specifies when the coroutine is resumed. Useful types of yields:
  - yield return new WaitForSeconds(float waitTime)
  - yield return null
- You can also stop coroutines

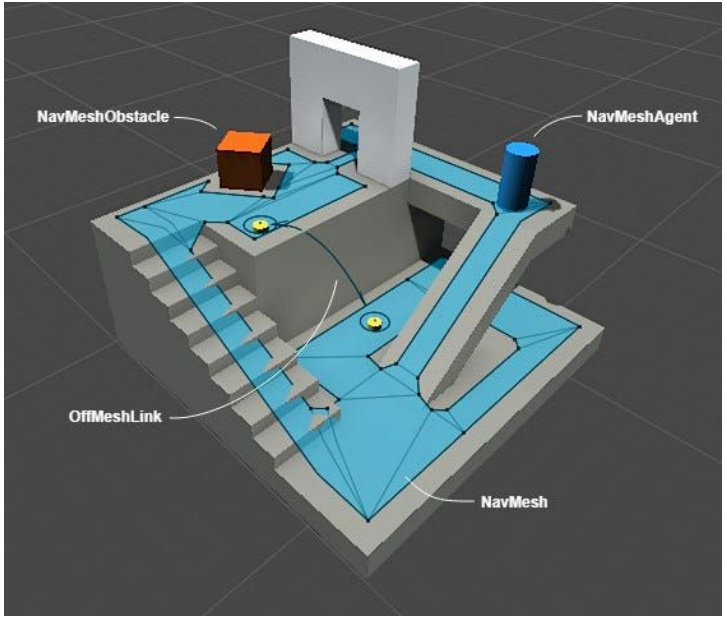
Coroutines

```
Event function
private void Start()
{
    StartCoroutine(routine: Patrol());
}

Frequently called  2 usages
private IEnumerator Patrol()
{
    var targetPos :Vector3 = patrolPoints[currentPoint].position;

    while (Vector3.Distance(a: transform.position, b: targetPos) > 0.1f)
    {
        // Move towards target position
        yield return null;
    }
    yield return new WaitForSeconds(waitTime);
    // update target position
    StartCoroutine(routine: Patrol());
}
```

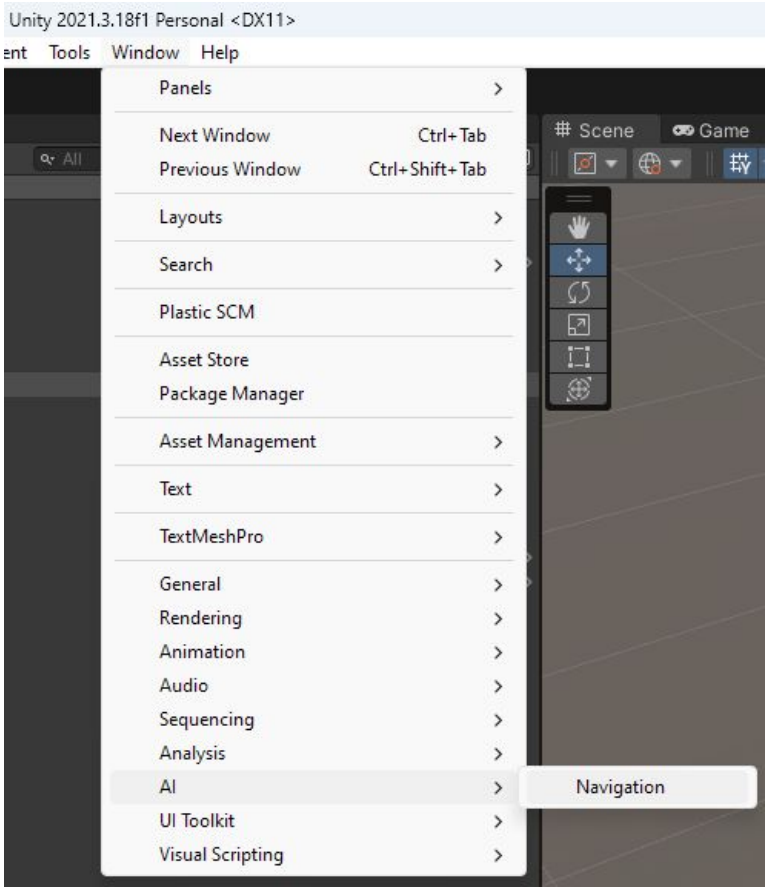
NavMesh

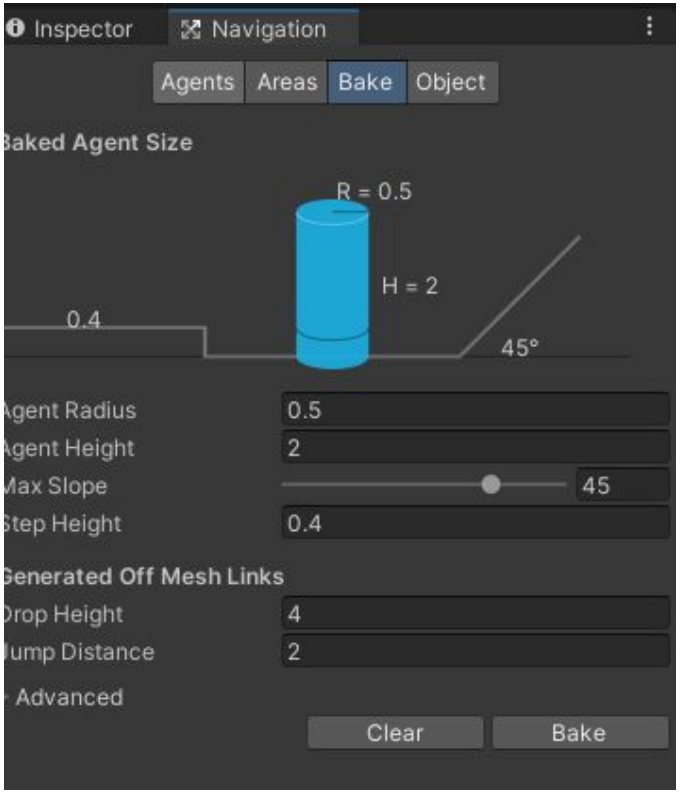
Unity’s built-in pathfinding solution

Bake a Navigation Mesh and your agents will traverse it

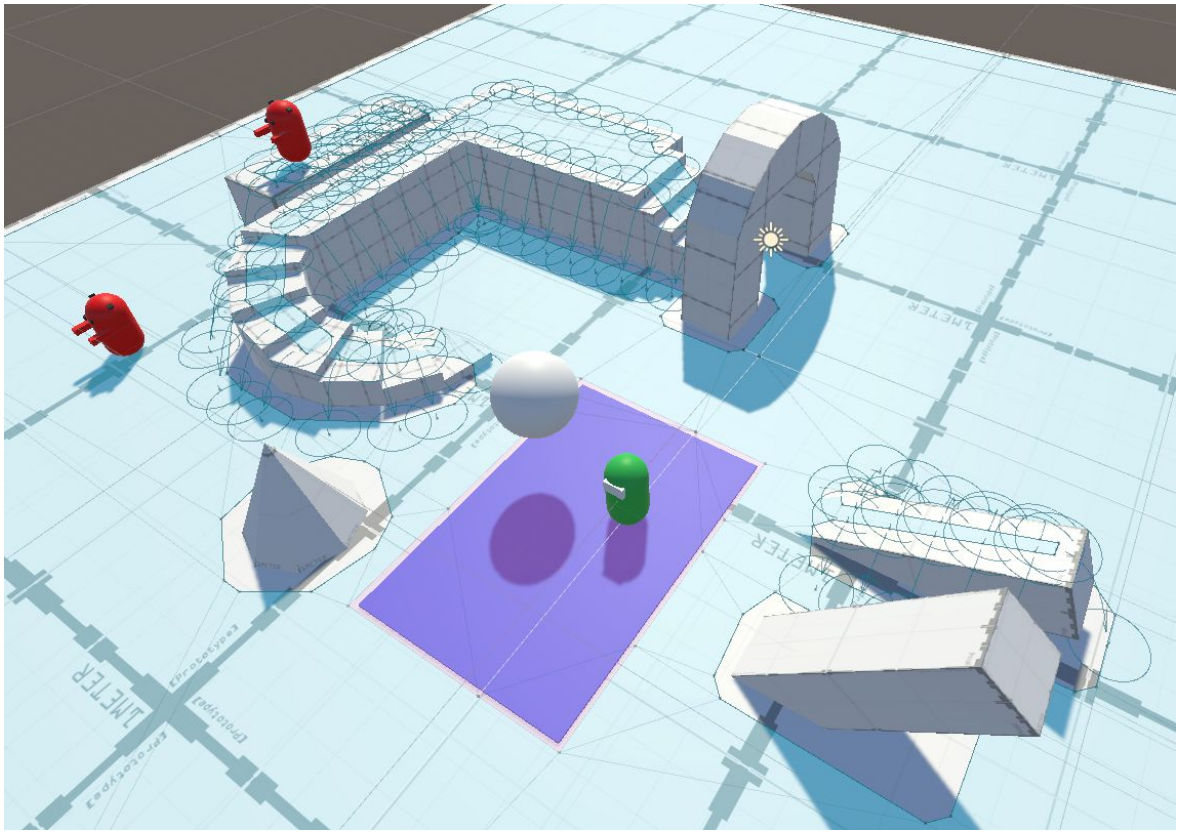


Navigation Window



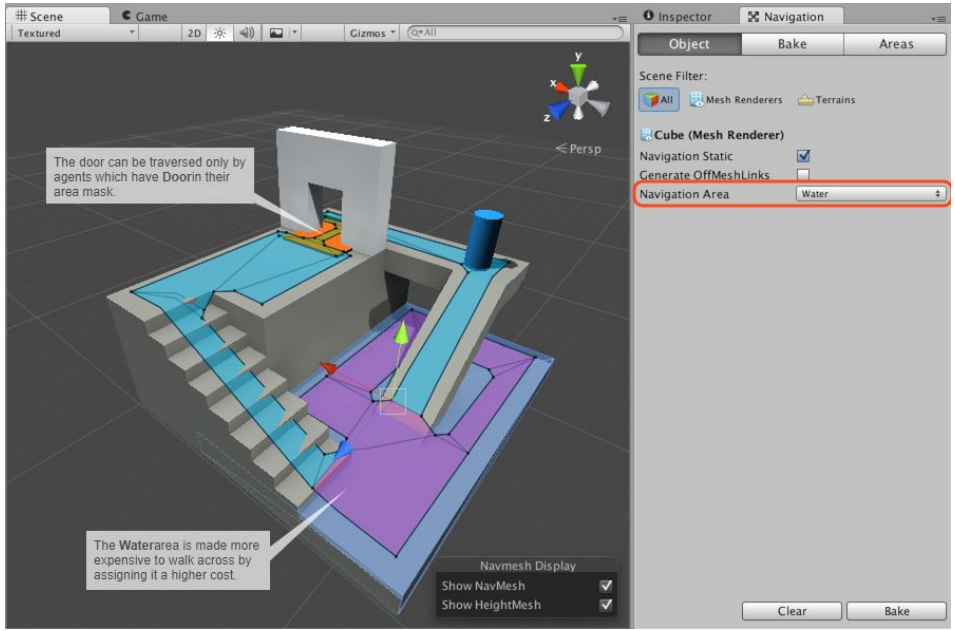


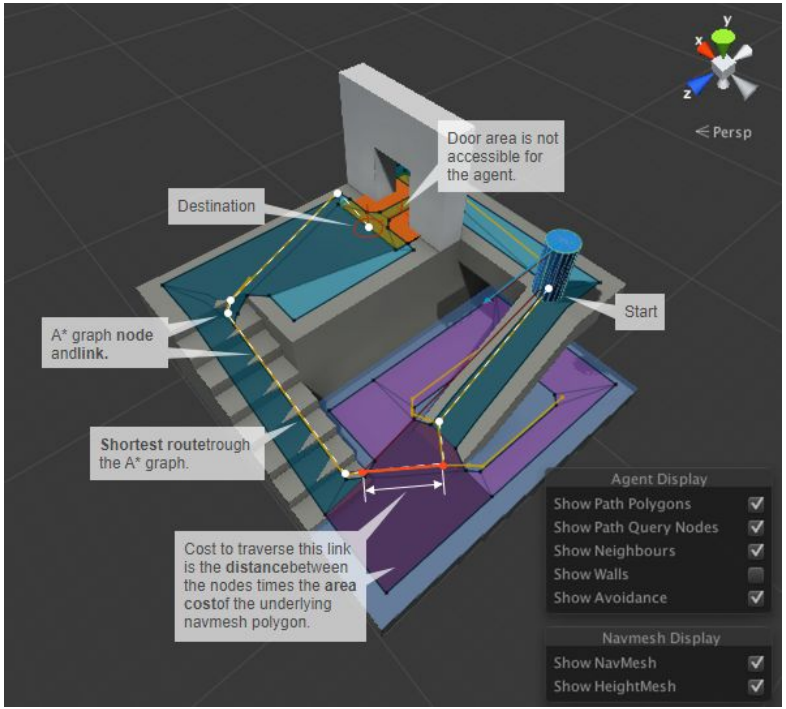
NavMesh Example



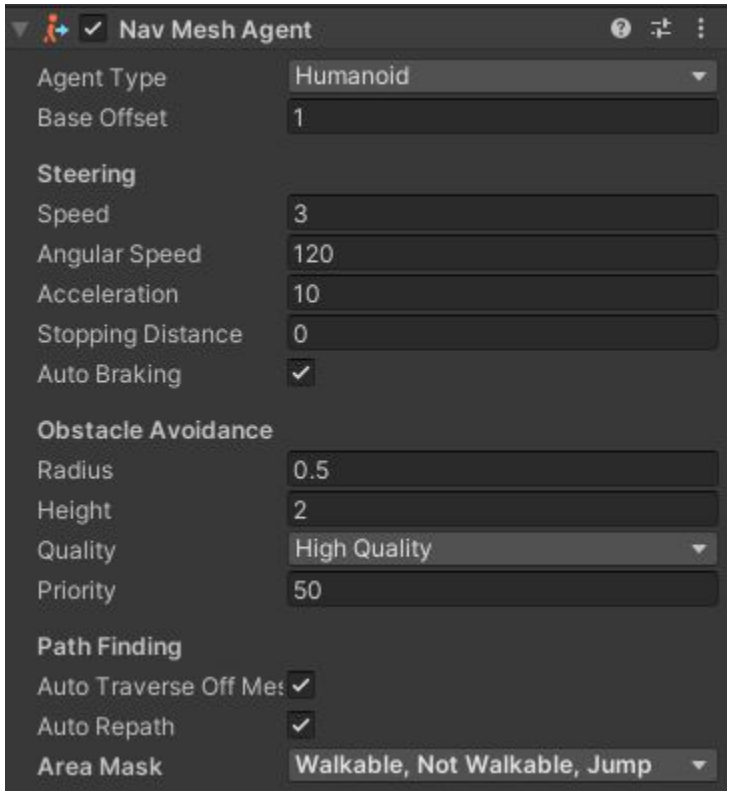
NavMesh Areas

- You can define what agents can traverse which areas
- A cost is associated with each area





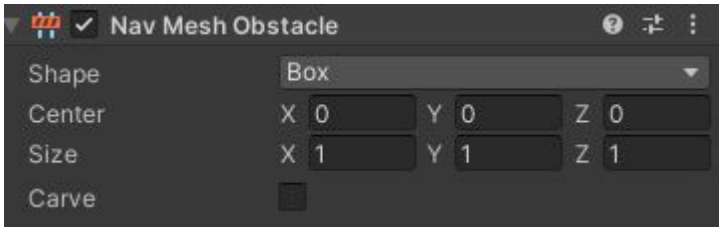
NavMesh Agent

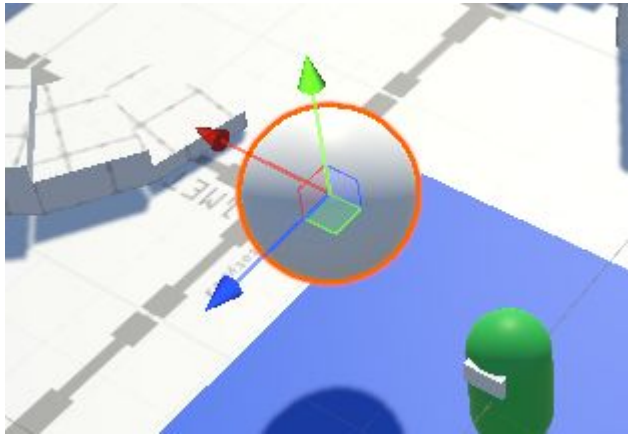



```
[SerializeField] private Transform target;
private NavMeshAgent agent;

private void Awake()
{
    agent = GetComponent<NavMeshAgent>();
}

private void Update()
{
    agent.destination = target.position;
}
```

NavMesh Obstacles

Allows for dynamic objects that the agents can account for in their pathing





Exercises

- More examples of AI behaviour with Unity NavMesh
- Full course on AI in Unity (some of this may be usable in your project):
  - Covers pathfinding, State Machines, Crowd Simulation, Behavioural Trees and more…

Train like an ML-Agent! ;)