# Ejemplos peatones

## RULEController.h

Clase abstracta pura (*mmm creo que no*): agentInstance tiene un rulecontroller.

Sistemas de reglas de **acciones** y **condiciones**

Texto

Descripción generada automáticamente

Bool all\_rules: evalua todas las reglas si es cierto.

## Rc\_boids

Tiene las 3 condiciones y acciones

Texto

Descripción generada automáticamente

Sp.QueryColserThan 🡪 mira a ver si hay alguien mas cerca de X metros

Matrices.cpp: tiene el initApp

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

El ultimo numero es el num de celdas.(si le pongo 4 va mucho más lento).

## SpatialIndex

Dibujo de un pizarrón blanco

Descripción generada automáticamente con confianza media

Mira quien está cerca. Si hay muchas celdas te ahorras muuuchos cálculos.

# Ejemplo Behaviour Trees

Bt 🡪 clase abstracta

BT\_soldier hereda de BT

**BT\_node:** motor de recursión.

* Nombre
* Tipo (random, secuencia, prioridad o accion)
* Vector de hijos
* Ptr a parent
* Ptr a rightSibling

**BT.h**:

* Map <string, bt\_node> 🡪 puedo encontrar un nodo por nombre (es el árbol) tree
* Map actions
* Map conditions

Texto

Descripción generada automáticamente

Un dibujo de un pizarrón blanco

Descripción generada automáticamente con confianza media

**UNREAL**

**Nodo paralelo**: ejecuta acciones a la vez

**Nodo decorador**: Hemos dicho que solo los nodos Prioritatorios tienen condiciones. Un nodo **decorador** es separar la condición a un nodo que sirve de filtro.

Solo tiene un hijo y devuelvo cierto o falso (solo si es cierto sigue). Y cualquier nodo puede tener un decorador.

En vez de hacer un random equiprobable pondré un DECORADOR que ponga un 90% de veces un nodo y un 10% el otro, por ejemplo.

Los códigos de retorno ahora se llama SUCCESS / FAIL

Decorators: condiciones. Puedo desactivar arboles enteros. Puede servir para distintas fases de un boss.

Composite: es lo que llamamos sequence or selector. (nodo con varios hijos).

Pantalla de juego de computadora

Descripción generada automáticamente con confianza media

UDN = UNREAL DEVELOPER NETWORK (Foros de unreal).

Simple Parallel: solo dos hijos y modo maestro/esclavo. Sus hijos son tareas.

* El izq es master (tiene que tener return de SUCCESS o FAIL)
* El drcho. Slave (Este solo para cuando pare el izquierdo).

Selector devuelve true si algún hijo se ejecuta.