12.4.1 Game Engine - Logic - Introduction

Logic

- Introduction
  - Logic Bricks
  - Properties
  - States
- Sensors
  - Introduction
  - Sensor Editing
  - Sensor Common Options
  - Sensor Types
- Controllers
  - Introduction
  - Controller Editing
  - Controller Types
- Actuators
  - Introduction
  - Actuator Editing
  - Actuator Common Options
  - Actuator Types
- Properties
  - Property Types
- States
  - How States Operate
  - Editing States

Introduction

Game Logic is the default scripting layer in the game engine. Each Game Object in the game may store a collection of logical components (Logic Bricks) which control its behavior within the scene. Logic bricks can be combined to perform user-defined actions that determine the progression of the simulation.

Logic Bricks

The main part of game logic can be set up through a graphical interface the *Logic Editor*, and therefore does not require detailed programming knowledge. Logic is set up as blocks (or “bricks”) which represent preprogrammed functions; these can be tweaked and combined to create the game/application. There are three types of logic brick: *Sensors*, *Controllers* and *Actuators*. Sensors are primitive event listeners, which are triggered by specific events, such as a collision, a key press or mouse movement. Controllers carry out logic operations on sensor output, and trigger connected actuators when their operating conditions are met. Actuators interact with the simulation directly, and are the only components in the game which are able to do so (other than the Python controller, and other simulation components such as Physics

Properties

Properties are like variables in other programming languages. They are used to save and access data values either for the whole game (eg. scores), or for particular objects/players (e.g. names). However, in the Blender Game Engine, a property is associated with an object. Properties can be of different types, and are set up in a special area of the *Logic Editor.*

States

Another useful feature is object *States*. At any time while the simulation is running, the object will process any logic which belongs to the current state of the object. States can be used to define groups of behavior - eg. an actor object may be “sleeping”, “awake” or “dead”, and its logic behavior may be different in each of these three states. The states of an object are set up, displayed and edited in the Controller logic bricks for the object.