Game Engine

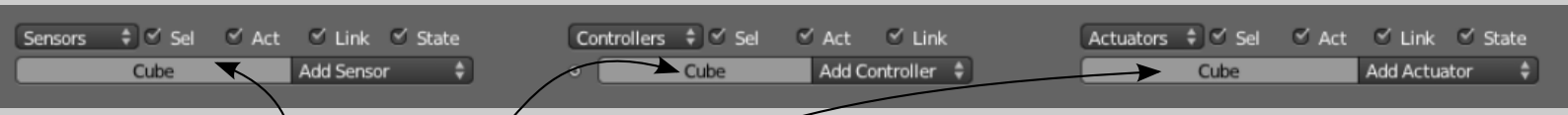
Sensors, Controllers & Actuators

We use sensors, controllers and actuators to create interactivity in our games. These logic bricks have been created so that you don't have to program any code.

Sensors "sense" things, such as nearby objects, a key pressed on the keyboard, or timed events. When a sensor is triggered a pulse is sent to all linked controllers.

Controllers handle the logic, evaluating pulses from sensors and sends the pulses to actuators in response.

Actuators affect objects or game in some way. They change motion, sound, properties, objects, etc. These changes can be in other objects, physics, properties or they can cause trigger events for other logic bricks.



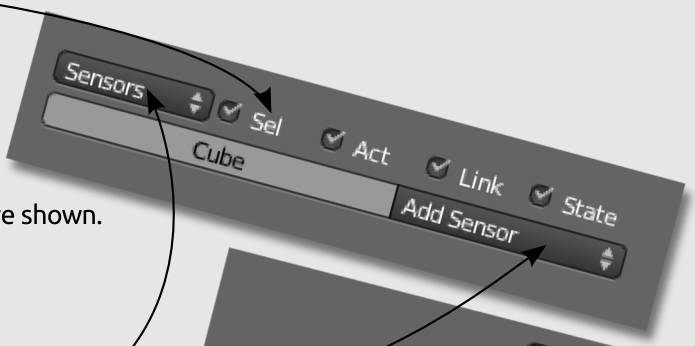
The name of the selected object that you will add the logic bricks to.

Sensors

Sensors are the things that make logic to do something.

At the top of the Sensors sub-panel, there are four buttons: Sel, Act, Link and State. These filter which sensors can be viewed.

Sel – Shows all selected objects sensors.
Act – Shows only active objects sensors.
Link – Shows sensors which have a link to a controller.
State – Only sensors connected to a controller of the current state are shown.



You must have a selected object in the 3D scene to apply a sensor.

There are two buttons, **Sensors**, and **Add Sensor**.

When you click on the **Add Sensor** button, a menu pops up. Each of these sensors does something different.

Sensor types

Actuator sensor
This sensor is set off when an actuator with the name specified in the Actuator field is activated.

Always sensor
This sensor is used for things that need to be done every logic tick, or every n logic tick (with the Frequency, ... tick down, tick up), or at start-up with Tap. There are 60 logic ticks pers second.

Collision sensor
This works like a Touch sensor, but can also filter by property. Only objects with the property with that name will generate a positive pulse upon collision. Leave blank for collision with any object. The pulse button makes it sensitive to other collisions even if it is still in touch with the object that triggered the last positive pulse. The M/P button toggles between material and property filtering.

Delay sensor
This sensor is designed for delaying reactions a number of logicticks. This is useful if another action has to be done first or totime events. Delay – is the number of logic ticks the sensor waits before sending a positive pulse. Duration – is the time the sensor waits before sending a negativepulse. Repeat – makes the sensor restart after the delay and durationtime is up.

Joystick sensor
This sensor detects joystick events. We won't be exploring this sensor in class.

Keyboard sensor
This sensor is for detecting keyboard input, it can also save keyboard input into a String property. The Key field is for single key presses. Press the button with no label to assign a key to the sensor. This is the active key. Pressing the key will trigger a positive pulse. The All Keys button will hide all other buttons and send positive pulses on all key presses. This isuseful for custom key maps with a Python controller. The Modifier buttons work the same ways as the key button, but won't set off a positive pulse unlessthey are held down while the key button is pressed. This is helpful if you want to use keys like CTRL+R or SHIFT+ALT+T to do specific action. Log Toggle assigns a Boolean property which determines if the keystroke will be logged in the Stringproperty (true) or not (false). This needs to be here if you want to log your keystrokes.Target is the String property in which the keystrokes are saved. Together with a Property sensor thiscan be used for example to enter passwords.

Message sensor
Messages can be used to send either text or property values. The message sensor sends a positive pulse once a message is sent anywhere in the engine, it can filter to only send a pulse upon a message with a specific subject.

Mouse sensor
This sensor is for detecting mouse input. It consist of a list of mouse inputs, all self-explanatory.

Near sensor
This sensor detects objects that are within a specific distance of themselves. It can filter objects with properties, like the Collision sensor. The Property field can be used to limit the sensor to look for only objects with this property. Distance is the number of blender units it will detect objects within. Reset Distance is the distance an object needs to be to reset the sensor (send a negative pulse).

Property sensor
This sensor detects changes in the object's properties. This bsensor has four modes:
Equal – triggers a positive pulse when the property values match the value in the sensor. The Property field is the name for the property, and Value is for the value it has to match and send a positive pulse.
Not equal – triggers a positive pulse when the property value differs from the value in the sensor.
Interval – triggers a positive pulse when the value of the property is between the Min and Max values of the sensor. For "more than", enter the property name in the Max field and the lowest number in the Min field. For "less than", enter the property name in the Min field and the maximum value in the Max field. Names of properties can be entered to compare properties.
Changed – sends a positive pulse as soon as the property value changes.

Radar sensor
This sensor works like a Near sensor, but only within an angle from an axis, forming an invisible conewith the top in the object's centre and base at a distance on an axis. The Property field can be used to limit the sensor to look for only the objects with this property.The Axis menu, determines the direction of the radar cone. The + or – signs determines the axisdirection, + is along the positive axis, - is along the negative axis.Angle determines the width of the cone.Distance determines the length of the cone.This sensor is useful for giving bots sight only in front of them. Note: it does see though other objects.

Random sensor
This sensor generates random pulses. It has a Seed field to enter the initial seed. 0 is not random, for testing and debugging purposes.

Ray sensor
This sensor shoots in the direction of an axis and sends a positive pulse once it hits something. It can be filtered to only detect objects with a given material or property. It shares a lot of buttons and fields with the Radar sensor. The X-Ray Mode button makes it x-ray, it sees through objects that don't have the property or material specified in the filter field.

Touch sensor
This sensor sends a positive pulse when the object is in contact with another object. The MA field is for filtering materials. Only contact with the material in this field will generate a positive pulse. Leave blank for touch with any object. The positive pulse is sent on collision and the negative pulse is sent once the objects are no longer in contact. For a continuous pulse while they are in contact use "true Pulse triggering".

Controller types

Controllers are the bricks that collect data sent by the sensors.
When a sensor is activated it sends out a positive pulse and when it is deactivated it sends out a negative pulse. The controller's job is to check and combine these pulses to trigger the proper response. There are eight ways to process the input:

AND, OR, XOR, NAND, NOR XNOR, Expression and Python.

In this information we will explore the AND and OR controllers, if you require more information about the other controllers please look online.

AND Controller
The AND controller is the default type when you create a new controller. This is because it is the most common controller and it works well for just passing a sensor directly to an actuator (which cannot be done without a controller).

An AND controller activates the actuators it is connected to only if all the sensors connected to it are positive. If only one sensor is connected to it, the actuators will be activated as soon as the sensor is triggered. If more sensors are connected they all have to be activated at the same time. This is where the name comes from. If there are two sensors, "sensor1" and "sensor2", connected to the controller both sensor1 AND sensor2 have to be active at the same time.

Example
You want to make a menu button. When the player clicks the button he is transferred to the next scene.

On the menu button you set up two Mouse Sensors, a Mouse over sensor and a Left button sensor. Both are connected to the same AND controller, which is connected to a Set Scene actuator. This means the player has to both hover the mouse over the menu button and click to get to the next scene.

OR Controller
An OR controller activates the actuators it is connected to if at least one of its sensors is activated. If just one sensor is connected to the controller it will pass on the positive pulse when that sensor is activated, if more sensors are connected, only one of them need to be activated. OR is not exclusive, which means it works like an AND controller too: if all sensors are activated the OR controlle will still pass the pulse. XOR controller is an exclusive or controller.

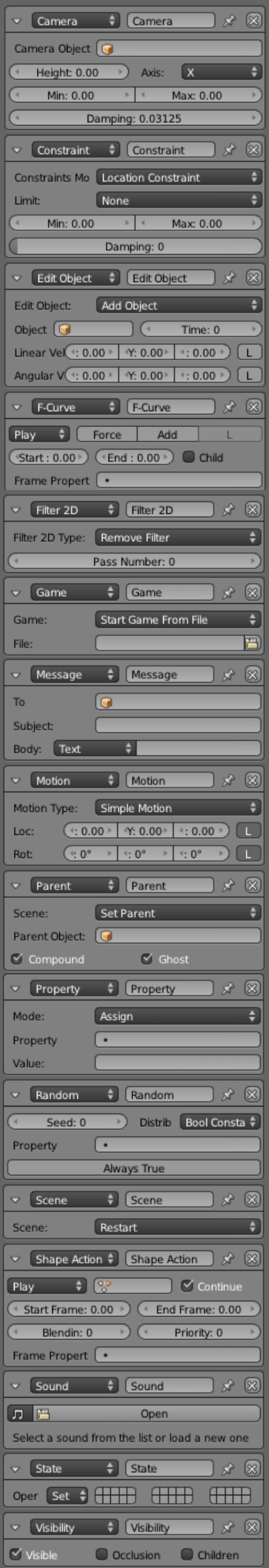
Example
You want both the Esc and the Q keys to quit your game. You set up two Keyboard sensors, one with Esc key and one with Q key. Both are then connected to an OR controller which is connected to a Quit this game actuator. If one of the two sensors is activated the game will then quit.

Expression Controller
This is a very confusing controller, but is really helpful in special cases.
You can customise your controller by typing into the field the name of the property and the action you want it to do.

Python Controller
The Python controller simply runs a loaded Python script.

Actuator types

Actuators perform actions, such as move, create (or destroy) objects or play a sound. They are initiated by the positive pulse from one (or more) controllers.



Camera
Has options to follow objects smoothly, primarily for camera objects but any object can use this.

Constraint
Used to limit an objects location, distance or rotation. These are useful for controlling the physics of the object in game.

Edit Object
Edits the object's mesh, adds objects or destroys them. It can also change the mesh of an object.

F-Curve
Controls animation curves, these can move, rotate, scale,change colour of objects and more.

2D Filters
Filters for special effects like sepia colours or motion blur.

Game
Handles the entire game and can do things as restart, quit, load and save.

Message
Sends messages, which can be received by other objects to activate them.

Motion
Sets object into motion and/or rotation, there are different options from "teleporting" to physically push, or rotate objects.

Parent
Can set a parent to the object, or unparent it.

Property
Manipulates the object's properties, like assigning, adding or copying.

Random
Creates random values which can be stored in properties.

Scene
Manage the scenes in your file, these can be used as levels or for UI and background.

Shape Action
Handles animations stored in shape key and animated with shape actions.

Sound
Used to play sounds in the game.

State
Changes states of the object.

Visibility
Changes visibility of the object.