GameLogic Class Index

[Index] [Hierarchy]

GameLogic	This is in fact no class it is a module in Python
PythonController	A PythonController is a special controller in Blender which can have a Python script bound to it
GameKeys	This is in fact no class it is a module in Python
Rasterizer	This is in fact no class it is a module in Python
Sensor	Baseclass for all sensors
Actuator	Baseclass for all actuators
TouchSensor	A TouchSensor is a special kind of a general sensor
NearSensor	A NearSensor is a special kind of a TouchSensor
RadarSensor	A RadarSensor is a special kind of a NearSensor
PropertySensor	A PropertySensor is a special kind of a general sensor
AlwaysSensor	A AlwaysSensor is a special kind of a general sensor
MouseSensor	A MouseSensor is a special kind of a general sensor
RandomSensor	A RandomSensor is a special kind of a general sensor
RaySensor	A RaySensor is a special kind of a general sensor
KeyboardSensor	A KeyboardSensor is a special kind of a general sensor
EndObjectActuator	A EndObjectActuator is a special kind of a general actuator
TrackToActuator	A TrackToActuator is a special kind of a general actuator
RandomActuator	A RandomActuator is a special kind of a general actuator
ReplaceMeshActuator	A ReplaceMeshActuator is a special kind of a general actuator
ConstraintActuator	A ConstraintActuator is a special kind of a general actuator
SoundActuator	A SoundActuator is a special kind of a general actuator
PropertyActuator	A PropertyActuator is a special kind of a general actuator
SceneActuator	A SceneActuator is a special kind of a general actuator
CameraActuator	A CameraActuator is a special kind of a general actuator
AddObjectActuator	A AddObjectActuator is a special kind of a general actuator
GroupActuator	A GroupActuator is a special kind of a general actuator
ObjectActuator	A ObjectActuator is a special kind of a general actuator
IpoActuator	A IpoActuator is a special kind of a general actuator
Object	Baseclass for all game objects
GameObject	A GameObject is a special kind of a general object in Blender's game engine
Camera	A Camera is a special kind of a GameObject in Blender's game engine
LightObject	A LightObject is a special kind of a GameObject in Blender's game engine
EmptyObject	A EmptyObject is a special kind of a GameObject in Blender's game engine

Controller	Baseclass for all controllers
ORController	A ORController is a special kind of a general controller
ANDController	A ANDController is a special kind of a general controller
LogicBrick	Baseclass for all sensors, controller, and actuators

---

GameLogic Class Hierarchy

- GameKeys
- GameLogic
- LogicBrick
    - Actuator
        - AddObjectActuator
        - CameraActuator
        - ConstraintActuator
        - EndObjectActuator
        - GroupActuator
        - IpoActuator
        - ObjectActuator
        - PropertyActuator
        - RandomActuator
        - ReplaceMeshActuator
        - SceneActuator
        - SoundActuator
        - TrackToActuator
    - Controller
        - ANDController
        - ORController
        - PythonController
    - Sensor
        - AlwaysSensor
        - KeyboardSensor
        - MouseSensor
        - PropertySensor
        - RandomSensor
        - RaySensor
        - TouchSensor

        - NearSensor
            - RadarSensor
- Object
    - GameObject
        - Camera
        - EmptyObject
        - LightObject
- Rasterizer

GameKeys Class Reference

[GameLogic Index] [GameLogic Hierarchy]

This is in fact no class it is a module in Python More...

`#include <gamekeys.h>`

Public Members

- **GameKeys** ()
- **~GameKeys** ()

Detailed Description

This is in fact no class it is a module in Python. It defines the constants "AKEY", ..., "ZKEY", "ZERO_KEY" , ..., "NINEKEY", "CAPSLOCKKEY", "LEFTCTRLKEY", "LEFTALTKEY", "RIGHTALTKEY", "RIGHTCTRLKEY", "RIGHTSHIFTKEY", "LEFTSHIFTKEY", "ESCKEY", "TABKEY", "RETKEY", "SPACEKEY", "LINEFEEDKEY", "BACKSPACEKEY", "DELKEY", "SEMICOLONKEY", "PERIODKEY", "COMMAKEY", "QUOTEKEY", "ACCENTGRAVEKEY", "MINUSKEY", "VIRGULEKEY", "SLASHKEY", "BACKSLASHKEY", "EQUALKEY", "LEFTBRACKETKEY", "RIGHTBRACKETKEY", "LEFTARROWKEY", "DOWNARROWKEY", "RIGHTARROWKEY", "UPARROWKEY", "PAD0", ..., "PAD9", "PADPERIOD", "PADVIRGULEKEY", "PADASTERKEY", "PADMINUS", "PADENTER", "PADPLUSKEY", "F1KEY", ..., "F12KEY", "PAUSEKEY", "INSERTKEY", "HOMEKEY", "PAGEUPKEY", "PAGEDOWNKEY", and "ENDKEY".

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

GameLogic Class Reference

[GameLogic Index] [GameLogic Hierarchy]

This is in fact no class it is a module in Python More...

`#include <gamelogic.h>`

Public Members

- **GameLogic** ()
- **~GameLogic** ()
- PyObject* **getCurrentController** ()
- PyObject* **addActiveActuator** (PyObject* actuator, int boolean)

Detailed Description

This is in fact no class it is a module in Python. But I use it here as a class to document the global functions availabe in the GameLogic module.

PyObject* getCurrentController()

Gives Python access to the PythonController which is connected to the Python script currently running.

PyObject* addActiveActuator(PyObject* actuator, int boolean)

Decides if an actuator linked to the PythonController should be active or not (bool value).

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

LogicBrick Class Reference

[GameLogic Index] [GameLogic Hierarchy]

Baseclass for all sensors, controller, and actuators More...

`#include <logicbrick.h>`

Public Members

- **LogicBrick** ()
- **~LogicBrick** ()
- PyObject* **getOwner** ()

Detailed Description

Baseclass for all sensors, controller, and actuators.

PyObject* getOwner()

Returns the owner of the Python script.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

Actuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

Baseclass for all actuators More...

`#include <actuator.h>`

Inherits: LogicBrick

Public Members

- **Actuator** ()
- **~Actuator** ()

Detailed Description

Baseclass for all actuators.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

AddObjectActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A AddObjectActuator is a special kind of a general actuator More...

`#include <addobjectactuator.h>`

Inherits: Actuator

Public Members

- **AddObjectActuator** ()
- **~AddObjectActuator** ()
- PyObject* **setObject** (char* name)
- PyObject* **getObject** ()
- PyObject* **setTime** (int duration)
- PyObject* **getTime** ()
- PyObject* **setLinearVelocity** (float vx, float vy, float vz)
- PyObject* **getLinearVelocity** ()

Detailed Description

A AddObjectActuator is a special kind of a general actuator.

PyObject* setObject(char* name)

Sets the object that will be added. There has to be an object of this name. If not, this function does nothing.

PyObject* getObject()

Returns the name of the object that will be added.

PyObject* setTime(int duration)

Sets the lifetime of the object that will be added, in frames. If the duration is negative, it is set to 0.

PyObject* getTime()

Returns the lifetime of the object that will be added.

PyObject* setLinearVelocity(float vx, float vy, float vz)

Assign this velocity to the created object.

PyObject* getLinearVelocity()

Returns the linear velocity that will be assigned to the created object.

CameraActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A CameraActuator is a special kind of a general actuator More...

`#include <cameraactuator.h>`

Inherits: Actuator

Public Members

- **CameraActuator** ()
- **~CameraActuator** ()

Detailed Description

A CameraActuator is a special kind of a general actuator.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

ConstraintActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A ConstraintActuator is a special kind of a general actuator More...

`#include <constraintactuator.h>`

Inherits: Actuator

Public Members

- **ConstraintActuator** ()
- **~ConstraintActuator** ()
- PyObject* **getMax** ()
- PyObject* **setMax** (float upper_bound)
- PyObject* **getMin** ()
- PyObject* **setMin** (float lower_bound)
- PyObject* **getDamp** ()
- PyObject* **setDamp** (int duration)
- PyObject* **setLimit** (int type)
- PyObject* **getLimit** ()

Detailed Description

A ConstraintActuator is a special kind of a general actuator.

PyObject* getMax()

Returns the upper value of the interval to which the value is clipped.

PyObject* setMax(float upper_bound)

Sets the upper value of the interval to which the value is clipped.

PyObject* getMin()

Returns the lower value of the interval to which the value is clipped.

PyObject* setMin(float lower_bound)

Sets the lower value of the interval to which the value is clipped.

PyObject* getDamp()

Returns the damping time for application of the constraint.

PyObject* setDamp(int duration)

Sets the time with which the constraint application is delayed. If the duration is negative, it is set to 0.

PyObject* setLimit(int type)

Sets the type of constraint. The variable type can be KX_CONSTRAINTACT_LOCX, KX_CONSTRAINTACT_LOCY, KX_CONSTRAINTACT_LOCZ, KX_CONSTRAINTACT_ROTX, KX_CONSTRAINTACT_ROTY, or KX_CONSTRAINTACT_ROTZ.

PyObject* getLimit()

Returns the type of constraint.

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

EndObjectActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A EndObjectActuator is a special kind of a general actuator More...

`#include <endobjectactuator.h>`

Inherits: Actuator

Public Members

- **EndObjectActuator** ()
- **~EndObjectActuator** ()

Detailed Description

A EndObjectActuator is a special kind of a general actuator.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

GroupActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A GroupActuator is a special kind of a general actuator More...

`#include <groupactuator.h>`

Inherits: Actuator

Public Members

- **GroupActuator** ()
- **~GroupActuator** ()

Detailed Description

A GroupActuator is a special kind of a general actuator.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

IpoActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A IpoActuator is a special kind of a general actuator More...

`#include <ipoactuator.h>`

Inherits: Actuator

Public Members

- **IpoActuator** ()
- **~IpoActuator** ()
- PyObject* **set** (char* mode, int startFrame, int stopFrame, char* forceToggle)
- PyObject* **setKey2Key** (char* previous, char* cycle, char* hold, char* propertyName)
- PyObject* **setProperty** (char* propertyName)

Detailed Description

A IpoActuator is a special kind of a general actuator.

PyObject* set(char* mode, int startFrame, int stopFrame, char* forceToggle)

Possible values for mode "PLAY", "PINGPONG", "FLIPPER", "LOOPSTOP", "LOOPEND".

PyObject* setKey2Key(char* previous, char* cycle, char* hold, char* propertyName)

bla

PyObject* setProperty(char* propertyName)

bla

- *Author*: Jan Walter
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

ObjectActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A ObjectActuator is a special kind of a general actuator More...

`#include <objectactuator.h>`

Inherits: Actuator

Public Members

- **ObjectActuator** ()
- **~ObjectActuator** ()
- PyObject* **getForce** ()
- PyObject* **setForce** (float x, float y, float z, int toggle)
- PyObject* **getTorque** ()
- PyObject* **setTorque** (float x, float y, float z, int toggle)
- PyObject* **getDLoc** ()
- PyObject* **setDLoc** (float x, float y, float z, int toggle)
- PyObject* **getDRot** ()
- PyObject* **setDRot** (float x, float y, float z, int toggle)
- PyObject* **getLinearVelocity** ()
- PyObject* **setLinearVelocity** (float x, float y, float z, int toggle)
- PyObject* **getAngularVelocity** ()
- PyObject* **setAngularVelocity** (float x, float y, float z, int toggle)

Detailed Description

A ObjectActuator is a special kind of a general actuator.

PyObject* getForce()

bla

PyObject* setForce(float x, float y, float z, int toggle)

bla

PyObject* getTorque()

bla

PyObject* setTorque(float x, float y, float z, int toggle)

bla

PyObject* getDLoc()

bla

PyObject* setDLoc(float x, float y, float z, int toggle)

bla

PyObject* getDRot()

bla

PyObject* setDRot(float x, float y, float z, int toggle)

bla

PyObject* getLinearVelocity()

bla

PyObject* setLinearVelocity(float x, float y, float z, int toggle)

bla

PyObject* getAngularVelocity()

bla

PyObject* setAngularVelocity(float x, float y, float z, int toggle)

bla

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

PropertyActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A PropertyActuator is a special kind of a general actuator More...

`#include <propertyactuator.h>`

Inherits: Actuator

Public Members

- **PropertyActuator** ()
- **~PropertyActuator** ()
- PyObject* **setProperty** (char* name)
- PyObject* **getProperty** ()
- PyObject* **setValue** (char* value)
- PyObject* **getValue** ()

Detailed Description

A PropertyActuator is a special kind of a general actuator.

PyObject* setProperty(char* name)

Set the property on which to operate. If there is no property of this name, the call is ignored.

PyObject* getProperty()

Return the property on which the actuator operates.

PyObject* setValue(char* value)

Set the value with which the actuator operates. If the value is not compatible with the type of the property, the subsequent action is ignored.

PyObject* getValue()

Returns the value with which the actuator operates.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

RandomActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A RandomActuator is a special kind of a general actuator More...

`#include <randomactuator.h>`

Inherits: Actuator

Public Members

- **RandomActuator** ()
- **~RandomActuator** ()
- PyObject* **setSeed** (int seed)
- PyObject* **getSeed** ()
- PyObject* **getPara1** ()
- PyObject* **getPara2** ()
- PyObject* **getDistribution** ()
- PyObject* **setProperty** (char* name)
- PyObject* **getProperty** (char* name)
- PyObject* **setBoolConst** (int value)
- PyObject* **setBoolUniform** ()
- PyObject* **setBoolBernouilli** (float value)
- PyObject* **setIntConst** (int value)
- PyObject* **setIntUniform** (int lower_bound, int upper_bound)
- PyObject* **setIntPoisson** (float value)
- PyObject* **setFloatConst** (float value)
- PyObject* **setFloatUniform** (float lower_bound, float upper_bound)
- PyObject* **setFloatNormal** (float mean, float standard_deviation)
- PyObject* **setFloatNegativeExponential** (float half_life)

Detailed Description

A RandomActuator is a special kind of a general actuator.

PyObject* setSeed(int seed)

Set the initial seed of the generator. Equal seeds produce equal series. If the seed is 0, the generator will produce the same value on every call.

PyObject* getSeed()

Returns the initial seed of the generator. Equal seeds produce equal series.

PyObject* getPara1()

Returns the first parameter of the active distribution. Refer to the documentation of the generator types for the meaning of this value.

PyObject* getPara2()

Returns the second parameter of the active distribution. Refer to the documentation of the generator types for the meaning of this value.

PyObject* getDistribution()

Returns the type of the active distribution.

PyObject* setProperty(char* name)

Set the property to which the random value is assigned. If the generator and property types do not match, the assignment is ignored.

PyObject* getProperty(char* name)

Return the property to which the random value is assigned. If the generator and property types do not match, the assignment is ignored.

PyObject* setBoolConst(int value)

Set this generator to produce a constant boolean value.

PyObject* setBoolUniform()

Set this generator to produce true and false, each with 50% chance of occuring.

PyObject* setBoolBernouilli(float value)

Return false value * 100% of the time.

PyObject* setIntConst(int value)

Always return value.

PyObject* setIntUniform(int lower_bound, int upper_bound)

Return a random integer between lower_bound and upper_bound. The boundaries are included.

PyObject* setIntPoisson(float value)

Return a Poisson-distributed number. This performs a series of Bernouilli tests with parameter value. It returns the number of tries needed to achieve succes.

PyObject* setFloatConst(float value)

Always return value.

PyObject* setFloatUniform(float lower_bound, float upper_bound)

Return a random integer between lower_bound and upper_bound.

PyObject* setFloatNormal(float mean, float standard_deviation)

Return normal-distributed numbers. The average is mean, and the deviation from the mean is characterized by standard_deviation.

PyObject* setFloatNegativeExponential(float half_life)

Return negative-exponentially distributed numbers. The half-life 'time' is characterized by half_life.

- *Author*: Jan Walter

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

ReplaceMeshActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A ReplaceMeshActuator is a special kind of a general actuator More...

`#include <replacemeshactuator.h>`

Inherits: Actuator

Public Members

- **ReplaceMeshActuator** ()
- **~ReplaceMeshActuator** ()
- PyObject* **setMesh** (char* name)

Detailed Description

A ReplaceMeshActuator is a special kind of a general actuator.

PyObject* setMesh(char* name)

Set the mesh that will be substituted for the current one.

- *Author*: Jan Walter
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

SceneActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A SceneActuator is a special kind of a general actuator More...

`#include <sceneactuator.h>`

Inherits: Actuator

Public Members

- **SceneActuator** ()
- **~SceneActuator** ()
- PyObject* **setUseRestart** (int flag)
- PyObject* **getUseRestart** ()
- PyObject* **setScene** (char* sceneName)
- PyObject* **getScene** ()
- PyObject* **setCamera** (char* camera)
- PyObject* **getCamera** ()

Detailed Description

A SceneActuator is a special kind of a general actuator.

PyObject* setUseRestart(int flag)

Set flag to 1 to restart the scene.

PyObject* getUseRestart()

Return whether the scene will be restarted.

PyObject* setScene(char* sceneName)

Set the scene to switch to.

PyObject* getScene()

Return the scene to switch to.

PyObject* setCamera(char* camera)

Set the camera to switch to.

PyObject* getCamera()

Return the camera to switch to.

- *Author*: Jan Walter

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

SoundActuator Class Reference

A SoundActuator is a special kind of a general actuator More...

`#include <soundactuator.h>`

Inherits: Actuator

Public Members

- **SoundActuator** ()
- **~SoundActuator** ()
- PyObject* **setFilename** (char* name)
- PyObject* **getFilename** ()

Detailed Description

A SoundActuator is a special kind of a general actuator.

PyObject* setFilename(char* name)

Filename of the sound file.

PyObject* getFilename()

Get filename of the sound file.

- *Author*: Jan Walter
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

TrackToActuator Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A TrackToActuator is a special kind of a general actuator More...

`#include <tracktoactuator.h>`

Inherits: Actuator

Public Members

- **TrackToActuator** ()
- **~TrackToActuator** ()
- PyObject* **getObject** ()
- PyObject* **setObject** (char* objectname)
- PyObject* **getTime** ()
- PyObject* **setTime** (int time)
- PyObject* **getUse3D** ()
- PyObject* **setUse3D** (int value)

Detailed Description

A TrackToActuator is a special kind of a general actuator.

PyObject* getObject()

Returns the object to track with the parent of this actuator.

PyObject* setObject(char* objectname)

Set the object to track with the parent of this actuator.

PyObject* getTime()

Return the time in frames with which the tracking motion is delayed.

PyObject* setTime(int time)

Set the time in frames with which to delay the tracking motion.

PyObject* getUse3D()

Returns 1 if the motion is allowed to extend in the z-direction.

PyObject* setUse3D(int value)

Set to 1 to allow the tracking motion to extend in the z-direction, set to 0 to lock the tracking motion to the x-y plane.

- *Author*: Jan Walter

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

Controller Class Reference

[GameLogic Index] [GameLogic Hierarchy]

Baseclass for all controllers More...

`#include <controller.h>`

Inherits: LogicBrick

Public Members

- **Controller** ()
- **~Controller** ()

Detailed Description

Baseclass for all controllers.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

ANDController Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A ANDController is a special kind of a general controller More...

`#include <andcontroller.h>`

Inherits: Controller

Public Members

- **ANDController** ()
- **~ANDController** ()

Detailed Description

A ANDController is a special kind of a general controller.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

ORController Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A ORController is a special kind of a general controller More...

`#include <orcontroller.h>`

Inherits: Controller

Public Members

- **ORController** ()
- **~ORController** ()

Detailed Description

A ORController is a special kind of a general controller.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

PythonController Class Reference

A PythonController is a special controller in Blender which can have a Python script bound to it More...

`#include <pythoncontroller.h>`

Inherits: Controller

Public Members

- **PythonController** ()
- **~PythonController** ()
- PyObject* **getActuators** ()
- PyObject* **getSensors** ()

Detailed Description

A PythonController is a special controller in Blender which can have a Python script bound to it. There are also sensors and actuators bound to this controller. The sensors are responsible for calling the Python script bound to the controller. The scripts "decides" how to react on this call and might call one of the actuators (or all) bound to the controller.

PyObject* getActuators()

Returns a Python list of actuators bound to the PythonController.

PyObject* getSensors()

Returns a Python list of sensors bound to the PythonController.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

Sensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

Baseclass for all sensors More...

`#include <sensor.h>`

Inherits: LogicBrick

Public Members

- **Sensor** ()
- **~Sensor** ()
- PyObject* **isPositive** ()
- PyObject* **getUsePosPulseMode** ()
- PyObject* **setUsePosPulseMode** (int flag)
- PyObject* **getPosFrequency** ()
- PyObject* **setPosFrequency** (int pulse_frequency)
- PyObject* **getUseNegPulseMode** ()
- PyObject* **setUseNegPulseMode** (int flag)
- PyObject* **getNegFrequency** ()
- PyObject* **setNegFrequency** (int pulse_frequency)
- PyObject* **getInvert** ()
- PyObject* **setInvert** (int invert)

Detailed Description

Baseclass for all sensors.

PyObject* isPositive()

Returns a Python integer.

PyObject* getUsePosPulseMode()

Returns TRUE if positive pulse mode is active, FALSE if positive pulse mode is not active.

PyObject* setUsePosPulseMode(int flag)

Set flag to TRUE to switch on positive pulse mode, FALSE to switch off positive pulse mode.

PyObject* getPosFrequency()

Return the frequency of the updates in positive pulse mode.

PyObject* setPosFrequency(int pulse_frequency)

Set the frequency of the updates in positive pulse mode. If the frequency is negative, it is set to 0.

PyObject* getUseNegPulseMode()

Returns TRUE if negative pulse mode is active, FALSE if negative pulse mode is not active.

PyObject* setUseNegPulseMode(int flag)

Set flag to TRUE to switch on negative pulse mode, FALSE to switch off negative pulse mode.

PyObject* getNegFrequency()

Return the frequency of the updates in negative pulse mode.

PyObject* setNegFrequency(int pulse_frequency)

Set the frequency of the updates in negative pulse mode. If the frequency is negative, it is set to 0.

PyObject* getInvert()

Returns whether or not pulses from this sensor are inverted.

PyObject* setInvert(int invert)

Set to TRUE to invert the responses of this sensor, set to KX_FALSE to keep the normal response.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

AlwaysSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A AlwaysSensor is a special kind of a general sensor More...

`#include <alwayssensor.h>`

Inherits: Sensor

Public Members

- **AlwaysSensor** ()
- **~AlwaysSensor** ()

Detailed Description

A AlwaysSensor is a special kind of a general sensor.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

KeyboardSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A KeyboardSensor is a special kind of a general sensor More...

`#include <keyboardsensor.h>`

Inherits: Sensor

Public Members

- **KeyboardSensor** ()
- **~KeyboardSensor** ()
- PyObject* **setKey** (int keycode)
- PyObject* **getKey** ()
- PyObject* **setHold1** (int keycode)
- PyObject* **getHold1** ()
- PyObject* **setHold2** (int keycode)
- PyObject* **getHold2** ()
- PyObject* **getPressedKeys** ()

Detailed Description

A KeyboardSensor is a special kind of a general sensor.

PyObject* setKey(int keycode)

Set the key this sensor should listen to. The variable keycode can be any code from GameKeys.

PyObject* getKey()

Return the code of the key this sensor is listening to.

PyObject* setHold1(int keycode)

Set the first modifier to the key this sensor should listen to. The variable keycode can be any code from GameKeys.

PyObject* getHold1()

Return the code of the first key modifier to the key this sensor is listening to.

PyObject* setHold2(int keycode)

Set the second modifier to the key this sensor should listen to. The variable keycode can be any code from GameKeys.

PyObject* getHold2()

Return the code of the second key modifier to the key this sensor is listening to.

PyObject* getPressedKeys()

Get a list of pressed keys that have either been pressed, or just released this frame.

MouseSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A MouseSensor is a special kind of a general sensor More...

`#include <mousesensor.h>`

Inherits: Sensor

Public Members

- **MouseSensor** ()
- **~MouseSensor** ()
- PyObject* **getXPosition** ()
- PyObject* **getYPosition** ()

Detailed Description

A MouseSensor is a special kind of a general sensor.

PyObject* getXPosition()

Returns the x-coordinate of the mouse sensor, in frame coordinates. The lower-left corner is the origin. The coordinate is given in pixels.

PyObject* getYPosition()

Returns the y-coordinate of the mouse sensor, in frame coordinates. The lower-left corner is the origin. The coordinate is given in pixels.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

PropertySensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A PropertySensor is a special kind of a general sensor More...

`#include <propertysensor.h>`

Inherits: Sensor

Public Members

- **PropertySensor** ()
- **~PropertySensor** ()
- PyObject* **getType** ()
- PyObject* **setType** (int type)
- PyObject* **getProperty** (char* name)
- PyObject* **setProperty** (char* name)
- PyObject* **getValue** ()
- PyObject* **setValue** (char* value)

Detailed Description

A PropertySensor is a special kind of a general sensor.

PyObject* getType()

Returns the type of check this sensor performs.

PyObject* setType(int type)

Set the type of check to perform. Valid arguments are KX_PROPSENSOR_EQUAL, KX_PROPSENSOR_NOTEQUAL, KX_PROPSENSOR_INTERVAL, KX_PROPSENSOR_CHANGED, or KX_PROPSENSOR_EXPRESSION.

PyObject* getProperty(char* name)

Return the property with which the sensor operates.

PyObject* setProperty(char* name)

Sets the property with which to operate. If there is no property of this name, the call is ignored.

PyObject* getValue()

Returns the value with which the sensor operates.

PyObject* setValue(char* value)

Set the value with which the sensor operates. If the value is not compatible with the type of the property, the subsequent action is ignored.

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

RandomSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A RandomSensor is a special kind of a general sensor More...

`#include <randomsensor.h>`

Inherits: Sensor

Public Members

- **RandomSensor** ()
- **~RandomSensor** ()
- PyObject* **setSeed** (int seed)
- PyObject* **getSeed** ()
- PyObject* **getLastDraw** ()

Detailed Description

A RandomSensor is a special kind of a general sensor.

PyObject* setSeed(int seed)

Set the initial seed of the generator. Equal seeds produce equal series. If the seed is 0, the generator will produce the same value on every call.

PyObject* getSeed()

Returns the initial seed of the generator. Equal seeds produce equal series.

PyObject* getLastDraw()

Return the last value that was drawn.

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

RaySensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A RaySensor is a special kind of a general sensor More...

`#include <raysensor.h>`

Inherits: Sensor

Public Members

- **RaySensor** ()
- **~RaySensor** ()
- PyObject* **getHitPosition** ()
- PyObject* **getHitObject** ()

Detailed Description

A RaySensor is a special kind of a general sensor.

PyObject* getHitPosition()

Returns the position where the object was hit by this ray.

PyObject* getHitObject()

Returns the name of the object that was hit by this ray.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

TouchSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A TouchSensor is a special kind of a general sensor More...

`#include <touchsensor.h>`

Inherits: Sensor

Public Members

- **TouchSensor** ()
- **~TouchSensor** ()
- PyObject* **setProperty** (char* name)
- PyObject* **getProperty** ()
- PyObject* **getHitObject** ()
- PyObject* **getHitObjectList** ()

Detailed Description

A TouchSensor is a special kind of a general sensor.

PyObject* setProperty(char* name)

Set the property or material to collide with. Use setTouchMaterial() to switch between properties and materials.

PyObject* getProperty()

Returns the property or material to collide with. Use getTouchMaterial() to find out whether this sensor looks for properties or materials.

PyObject* getHitObject()

Returns the Python object you touched.

PyObject* getHitObjectList()

Returns a Python list of touched objects.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

NearSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A NearSensor is a special kind of a TouchSensor More...

`#include <nearsensor.h>`

Inherits: TouchSensor

Public Members

- **NearSensor** ()
- **~NearSensor** ()

Detailed Description

A NearSensor is a special kind of a TouchSensor.

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

RadarSensor Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A RadarSensor is a special kind of a NearSensor More...

`#include <radarsensor.h>`

Inherits: NearSensor

Public Members

- **RadarSensor** ()
- **~RadarSensor** ()

Detailed Description

A RadarSensor is a special kind of a NearSensor.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

Object Class Reference

[GameLogic Index] [GameLogic Hierarchy]

Baseclass for all game objects More...

`#include <object.h>`

Public Members

- **Object** ()
- **~Object** ()
- PyObject* **setOrientation** (PyObject* pylist)
- PyObject* **getOrientation** ()

Detailed Description

Baseclass for all game objects.

PyObject* setOrientation(PyObject* pylist)

A list of lists is used to set the orientation matrix within Blender's game engine.

PyObject* getOrientation()

A list of lists is returned for the orientation matrix within Blender's game engine.

- *Author*: Jan Walter

K*doc*

- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

GameObject Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A GameObject is a special kind of a general object in Blender's game engine More...

`#include <gameobject.h>`

Inherits: Object

Public Members

- **GameObject** ()
- **~GameObject** ()
- PyObject* **setVisible** (int visible)
- PyObject* **setPosition** (PyObject* pylist)
- PyObject* **getPosition** ()
- PyObject* **getLinearVelocity** ()
- PyObject* **getVelocity** ()
- PyObject* **getMass** ()
- PyObject* **getReactionForce** ()
- PyObject* **restoreDynamics** ()
- PyObject* **suspendDynamics** ()

Detailed Description

A GameObject is a special kind of a general object in Blender's game engine.

PyObject* setVisible(int visible)

bla

PyObject* setPosition(PyObject* pylist)

Takes a Python list of three elements (x, y, z).

PyObject* getPosition()

Returns a Python list of three elements (x, y, z).

PyObject* getLinearVelocity()

Returns a Python list of three elements.

PyObject* getVelocity()

Returns a Python list of three elements.

PyObject* getMass()

bla

PyObject* getReactionForce()

Returns a Python list of three elements.

PyObject* restoreDynamics()

bla

PyObject* suspendDynamics()

bla

Camera Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A Camera is a special kind of a GameObject in Blender's game engine More...

`#include <camera.h>`

Inherits: GameObject

Public Members

- **Camera** ()
- **~Camera** ()

Detailed Description

A Camera is a special kind of a GameObject in Blender's game engine.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

EmptyObject Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A EmptyObject is a special kind of a GameObject in Blender's game engine More...

`#include <emptyobject.h>`

Inherits: GameObject

Public Members

- **EmptyObject** ()
- **~EmptyObject** ()

Detailed Description

A EmptyObject is a special kind of a GameObject in Blender's game engine.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

LightObject Class Reference

[GameLogic Index] [GameLogic Hierarchy]

A LightObject is a special kind of a GameObject in Blender's game engine More...

`#include <lightobject.h>`

Inherits: GameObject

Public Members

- **LightObject** ()
- **~LightObject** ()

Detailed Description

A LightObject is a special kind of a GameObject in Blender's game engine.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

Rasterizer Class Reference

[GameLogic Index] [GameLogic Hierarchy]

This is in fact no class it is a module in Python More...

`#include <rasterizer.h>`

Public Members

- **Rasterizer** ()
- **~Rasterizer** ()
- PyObject* **getWindowWidth** ()
- PyObject* **getWindowHeight** ()
- PyObject* **enableVisibility** (int visible)

Detailed Description

This is in fact no class it is a module in Python. But I use it here as a class to document the global functions availabe in the Rasterizer module.

PyObject* getWindowWidth()

Returns the window width as a Python integer.

PyObject* getWindowHeight()

Returns the window height as a Python integer.

PyObject* enableVisibility(int visible)

Enable or disable the visibility. The return value is None.

- *Author*: Jan Walter **K***doc*
- Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

```
/***************************************************************************
                          gamelogic.h  -  description
                             ------------------
    begin                : Thu Dec 14 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef GAMELOGIC_H
#define GAMELOGIC_H

struct PyObject;

/**This is in fact no class it is a module in Python. But I use it here as a class to
document the global functions availabe in the GameLogic module.
  *@author Jan Walter
  */

class GameLogic {
public:
        GameLogic();
        ~GameLogic();
  /** Gives Python access to the PythonController which is connected to the Python
script currently running. */
  PyObject* getCurrentController();
  /** Decides if an actuator linked to the PythonController should be active or not
(bool value). */
  PyObject* addActiveActuator(PyObject* actuator, int boolean);
};

#endif
```

Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

```
/***************************************************************************
                          pythoncontroller.h  -  description
                             ------------------
    begin                : Thu Dec 14 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef PYTHONCONTROLLER_H
#define PYTHONCONTROLLER_H

#include <controller.h>

struct PyObject;

/**A PythonController is a special controller in Blender which can have a Python
script bound to it. There are also sensors and actuators bound to this controller.
The sensors are responsible for calling the Python script bound to the controller.
The scripts "decides" how to react on this call and might call one of the actuators
(or all) bound to the controller.
  *@author Jan Walter
  */

class Controller.html">PythonController : public Controller  {
public:
        PythonController();
        ~PythonController();
  /** Returns a Python list of actuators bound to the PythonController. */
  PyObject* getActuators();
  /** Returns a Python list of sensors bound to the PythonController.
 */
  PyObject* getSensors();
};

#endif
```

```
/***************************************************************************
                          gamekeys.h  -  description
                             -------------------
    begin                : Thu Dec 14 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef GAMEKEYS_H
#define GAMEKEYS_H


/**This is in fact no class it is a module in Python.  It defines the constants
"AKEY", ..., "ZKEY", "ZERO_KEY" , ..., "NINEKEY", "CAPSLOCKKEY", "LEFTCTRLKEY",
"LEFTALTKEY", "RIGHTALTKEY", "RIGHTCTRLKEY", "RIGHTSHIFTKEY", "LEFTSHIFTKEY",
"ESCKEY", "TABKEY", "RETKEY", "SPACEKEY", "LINEFEEDKEY", "BACKSPACEKEY", "DELKEY",
"SEMICOLONKEY", "PERIODKEY", "COMMAKEY", "QUOTEKEY", "ACCENTGRAVEKEY", "MINUSKEY",
"VIRGULEKEY", "SLASHKEY", "BACKSLASHKEY", "EQUALKEY", "LEFTBRACKETKEY",
"RIGHTBRACKETKEY", "LEFTARROWKEY", "DOWNARROWKEY", "RIGHTARROWKEY", "UPARROWKEY",
"PAD0", ..., "PAD9", "PADPERIOD", "PADVIRGULEKEY", "PADASTERKEY", "PADMINUS",
"PADENTER", "PADPLUSKEY", "F1KEY", ..., "F12KEY", "PAUSEKEY", "INSERTKEY", "HOMEKEY",
"PAGEUPKEY", "PAGEDOWNKEY", and "ENDKEY".
  *@author Jan Walter
  */

class GameKeys {
public:
        GameKeys();
        ~GameKeys();
};

#endif
```

```
/***************************************************************************
                          rasterizer.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef RASTERIZER_H
#define RASTERIZER_H

struct PyObject;

/**This is in fact no class it is a module in Python. But I use it here as a class to
document the global functions availabe in the Rasterizer module.
  *@author Jan Walter
  */

class Rasterizer {
public:
        Rasterizer();
        ~Rasterizer();
  /** Returns the window width as a Python integer. */
  PyObject* getWindowWidth();
  /** Returns the window height as a Python integer. */
  PyObject* getWindowHeight();
  /** Enable or disable the visibility. The return value is None. */
  PyObject* enableVisibility(int visible);
};

#endif
```

```
/***************************************************************************
                          sensor.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef SENSOR_H
#define SENSOR_H

#include <logicbrick.h>

/**Baseclass for all sensors.
  *@author Jan Walter
  */

class Sensor : public LogicBrick  {
public:
        Sensor();
        ~Sensor();
  /** Returns a Python integer. */
  PyObject* isPositive();
  /** Returns TRUE if positive pulse mode is active, FALSE if positive pulse mode is
not active. */
  PyObject* getUsePosPulseMode();
  /** Set flag to TRUE to switch on positive pulse mode, FALSE to switch off positive
pulse mode. */
  PyObject* setUsePosPulseMode(int flag);
  /** Return the frequency of the updates in positive pulse mode. */
  PyObject* getPosFrequency();
  /** Set the frequency of the updates in positive pulse mode. If the frequency is
negative, it is set to 0. */
  PyObject* setPosFrequency(int pulse_frequency);
  /** Returns TRUE if negative pulse mode is active, FALSE if negative pulse mode is
not active. */
  PyObject* getUseNegPulseMode();
  /** Set flag to TRUE to switch on negative pulse mode, FALSE to switch off negative
pulse mode. */
  PyObject* setUseNegPulseMode(int flag);
  /** Return the frequency of the updates in negative pulse mode. */
  PyObject* getNegFrequency();
  /** Set the frequency of the updates in negative pulse mode. If the frequency is
negative, it is set to 0. */
  PyObject* setNegFrequency(int pulse_frequency);
  /** Returns whether or not pulses from this sensor are inverted. */
```

```
  PyObject* getInvert();
  /** Set to TRUE to invert the responses of this sensor, set to KX_FALSE to keep the
normal response. */
  PyObject* setInvert(int invert);
};

#endif
```

```
/***************************************************************************
                          actuator.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ACTUATOR_H
#define ACTUATOR_H

#include <logicbrick.h>

/**Baseclass for all actuators.
  *@author Jan Walter
  */

class Actuator : public LogicBrick  {
public:
        Actuator();
        ~Actuator();
};

#endif
```

Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

```
/***************************************************************************
                          touchsensor.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef TOUCHSENSOR_H
#define TOUCHSENSOR_H

#include <sensor.h>

/**A TouchSensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">TouchSensor : public Sensor  {
public:
        TouchSensor();
        ~TouchSensor();
  /** Set the property or material to collide with. Use setTouchMaterial() to switch
between properties and materials. */
  PyObject* setProperty(char* name);
  /** Returns the property or material to collide with. Use getTouchMaterial() to
find out whether this sensor looks for properties or materials. */
  PyObject* getProperty();
  /** Returns the Python object you touched. */
  PyObject* getHitObject();
  /** Returns a Python list of touched objects. */
  PyObject* getHitObjectList();
};

#endif
```

```
/***************************************************************************
                          nearsensor.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef NEARSENSOR_H
#define NEARSENSOR_H

#include <touchsensor.h>

/**A NearSensor is a special kind of a TouchSensor.
  *@author Jan Walter
  */

class NearSensor : public TouchSensor  {
public:
        NearSensor();
        ~NearSensor();
};

#endif
```

```
/***************************************************************************
                          radarsensor.h  -  description
                             -------------------
    begin                : Tue Dec 19 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef RADARSENSOR_H
#define RADARSENSOR_H

#include <nearsensor.h>

/**A RadarSensor is a special kind of a NearSensor.
  *@author Jan Walter
  */

class RadarSensor : public NearSensor  {
public:
        RadarSensor();
        ~RadarSensor();
};

#endif
```

```
/*****************************************************************************
                          propertysensor.h  -  description
                             ------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 *****************************************************************************/

/*****************************************************************************
 *                                                                           *
 *   This program is free software; you can redistribute it and/or modify    *
 *   it under the terms of the GNU General Public License as published by    *
 *   the Free Software Foundation; either version 2 of the License, or       *
 *   (at your option) any later version.                                     *
 *                                                                           *
 *****************************************************************************/

#ifndef PROPERTYSENSOR_H
#define PROPERTYSENSOR_H

#include <sensor.h>

/**A PropertySensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">PropertySensor : public Sensor  {
public:
        PropertySensor();
        ~PropertySensor();
  /** Returns the type of check this sensor performs. */
  PyObject* getType();
  /** Set the type of check to perform. Valid arguments are KX_PROPSENSOR_EQUAL,
KX_PROPSENSOR_NOTEQUAL, KX_PROPSENSOR_INTERVAL, KX_PROPSENSOR_CHANGED, or
KX_PROPSENSOR_EXPRESSION. */
  PyObject* setType(int type);
  /** Return the property with which the sensor operates. */
  PyObject* getProperty(char* name);
  /** Sets the property with which to operate. If there is no property of this name,
the call is ignored. */
  PyObject* setProperty(char* name);
  /** Returns the value with which the sensor operates. */
  PyObject* getValue();
  /** Set the value with which the sensor operates. If the value is not compatible
with the type of the property, the subsequent action is ignored. */
  PyObject* setValue(char* value);
};

#endif
```

```
/***************************************************************************
                          alwayssensor.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ALWAYSSENSOR_H
#define ALWAYSSENSOR_H

#include <sensor.h>

/**A AlwaysSensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">AlwaysSensor : public Sensor  {
public:
        AlwaysSensor();
        ~AlwaysSensor();
};

#endif
```

```
/***************************************************************************
                          mousesensor.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef MOUSESENSOR_H
#define MOUSESENSOR_H

#include <sensor.h>

/**A MouseSensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">MouseSensor : public Sensor  {
public:
        MouseSensor();
        ~MouseSensor();
  /** Returns the x-coordinate of the mouse sensor, in frame coordinates. The
lower-left corner is the origin. The coordinate is given in pixels. */
  PyObject* getXPosition();
  /** Returns the y-coordinate of the mouse sensor, in frame coordinates. The
lower-left corner is the origin. The coordinate is given in pixels. */
  PyObject* getYPosition();
};

#endif
```

```
/***************************************************************************
                          randomsensor.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef RANDOMSENSOR_H
#define RANDOMSENSOR_H

#include <sensor.h>

/**A RandomSensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">RandomSensor : public Sensor  {
public:
        RandomSensor();
        ~RandomSensor();
  /** Set the initial seed of the generator. Equal seeds produce equal series. If the
seed is 0, the generator will produce the same value on every call. */
  PyObject* setSeed(int seed);
  /** Returns the initial seed of the generator. Equal seeds produce equal series. */
  PyObject* getSeed();
  /** Return the last value that was drawn. */
  PyObject* getLastDraw();
};

#endif
```

```
/***************************************************************************
                          raysensor.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef RAYSENSOR_H
#define RAYSENSOR_H

#include <sensor.h>

/**A RaySensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">RaySensor : public Sensor  {
public:
        RaySensor();
        ~RaySensor();
  /** Returns the position where the object was hit by this ray. */
  PyObject* getHitPosition();
  /** Returns the name of the object that was hit by this ray. */
  PyObject* getHitObject();
};

#endif
```

```
/***************************************************************************
                          keyboardsensor.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef KEYBOARDSENSOR_H
#define KEYBOARDSENSOR_H

#include <sensor.h>

/**A KeyboardSensor is a special kind of a general sensor.
  *@author Jan Walter
  */

class Sensor.html">KeyboardSensor : public Sensor  {
public:
        KeyboardSensor();
        ~KeyboardSensor();
  /** Set the key this sensor should listen to. The variable keycode can be any code
from GameKeys. */
  PyObject* setKey(int keycode);
  /** Return the code of the key this sensor is listening to. */
  PyObject* getKey();
  /** Set the first modifier to the key this sensor should listen to. The variable
keycode can be any code from GameKeys. */
  PyObject* setHold1(int keycode);
  /** Return the code of the first key modifier to the key this sensor is listening
to. */
  PyObject* getHold1();
  /** Set the second modifier to the key this sensor should listen to. The variable
keycode can be any code from GameKeys. */
  PyObject* setHold2(int keycode);
  /** Return the code of the second key modifier to the key this sensor is listening
to. */
  PyObject* getHold2();
  /** Get a list of pressed keys that have either been pressed, or just released this
frame. */
  PyObject* getPressedKeys();
};

#endif
```

```
/***************************************************************************
                          endobjectactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ENDOBJECTACTUATOR_H
#define ENDOBJECTACTUATOR_H

#include <actuator.h>

/**A EndObjectActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">EndObjectActuator : public Actuator  {
public:
        EndObjectActuator();
        ~EndObjectActuator();
};

#endif
```

```
/***************************************************************************
                          tracktoactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef TRACKTOACTUATOR_H
#define TRACKTOACTUATOR_H

#include <actuator.h>

/**A TrackToActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">TrackToActuator : public Actuator  {
public:
        TrackToActuator();
        ~TrackToActuator();
  /** Returns the object to track with the parent of this actuator. */
  PyObject* getObject();
  /** Set the object to track with the parent of this actuator. */
  PyObject* setObject(char* objectname);
  /** Return the time in frames with which the tracking motion is delayed. */
  PyObject* getTime();
  /** Set the time in frames with which to delay the tracking motion. */
  PyObject* setTime(int time);
  /** Returns 1 if the motion is allowed to extend in the z-direction. */
  PyObject* getUse3D();
  /** Set to 1 to allow the tracking motion to extend in the z-direction, set to 0 to
lock the tracking motion to the x-y plane. */
  PyObject* setUse3D(int value);
};

#endif
```

```
/***************************************************************************
                          randomactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef RANDOMACTUATOR_H
#define RANDOMACTUATOR_H

#include <actuator.h>

/**A RandomActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">RandomActuator : public Actuator  {
public:
        RandomActuator();
        ~RandomActuator();
  /** Set the initial seed of the generator. Equal seeds produce equal series. If the
seed is 0, the generator will produce the same value on every call. */
  PyObject* setSeed(int seed);
  /** Returns the initial seed of the generator. Equal seeds produce equal series. */
  PyObject* getSeed();
  /** Returns the first parameter of the active distribution. Refer to the
documentation of the generator types for the meaning of this value. */
  PyObject* getPara1();
  /** Returns the second parameter of the active distribution. Refer to the
documentation of the generator types for the meaning of this value. */
  PyObject* getPara2();
  /** Returns the type of the active distribution. */
  PyObject* getDistribution();
  /** Set the property to which the random value is assigned. If the generator and
property types do not match, the assignment is ignored. */
  PyObject* setProperty(char* name);
  /** Return the property to which the random value is assigned. If the generator and
property types do not match, the assignment is ignored. */
  PyObject* getProperty(char* name);
  /** Set this generator to produce a constant boolean value. */
  PyObject* setBoolConst(int value);
  /** Set this generator to produce true and false, each with 50% chance of occuring.
*/
  PyObject* setBoolUniform();
  /** Return false value * 100% of the time. */
```

```
  PyObject* setBoolBernouilli(float value);
  /** Always return value. */
  PyObject* setIntConst(int value);
  /** Return a random integer between lower_bound and upper_bound. The boundaries are
included. */
  PyObject* setIntUniform(int lower_bound, int upper_bound);
  /** Return a Poisson-distributed number. This performs a series of Bernouilli tests
with parameter value. It returns the number of tries needed to achieve succes. */
  PyObject* setIntPoisson(float value);
  /** Always return value. */
  PyObject* setFloatConst(float value);
  /** Return a random integer between lower_bound and upper_bound. */
  PyObject* setFloatUniform(float lower_bound, float upper_bound);
  /** Return normal-distributed numbers. The average is mean, and the deviation from
the mean is characterized by standard_deviation. */
  PyObject* setFloatNormal(float mean, float standard_deviation);
  /** Return negative-exponentially distributed numbers. The half-life 'time' is
characterized by half_life. */
  PyObject* setFloatNegativeExponential(float half_life);
};

#endif
```

```
/***************************************************************************
                          replacemeshactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef REPLACEMESHACTUATOR_H
#define REPLACEMESHACTUATOR_H

#include <actuator.h>

/**A ReplaceMeshActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">ReplaceMeshActuator : public Actuator  {
public:
        ReplaceMeshActuator();
        ~ReplaceMeshActuator();
  /** Set the mesh that will be substituted for the current one. */
  PyObject* setMesh(char* name);
};

#endif
```

```
/***************************************************************************
                          constraintactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef CONSTRAINTACTUATOR_H
#define CONSTRAINTACTUATOR_H

#include <actuator.h>

/**A ConstraintActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">ConstraintActuator : public Actuator  {
public:
        ConstraintActuator();
        ~ConstraintActuator();
  /** Returns the upper value of the interval to which the value is clipped. */
  PyObject* getMax();
  /** Sets the upper value of the interval to which the value is clipped. */
  PyObject* setMax(float upper_bound);
  /** Returns the lower value of the interval to which the value is clipped. */
  PyObject* getMin();
  /** Sets the lower value of the interval to which the value is clipped. */
  PyObject* setMin(float lower_bound);
  /** Returns the damping time for application of the constraint. */
  PyObject* getDamp();
  /** Sets the time with which the constraint application is delayed. If the duration
is negative, it is set to 0. */
  PyObject* setDamp(int duration);
  /** Sets the type of constraint. The variable type can be KX_CONSTRAINTACT_LOCX,
KX_CONSTRAINTACT_LOCY, KX_CONSTRAINTACT_LOCZ, KX_CONSTRAINTACT_ROTX,
KX_CONSTRAINTACT_ROTY, or KX_CONSTRAINTACT_ROTZ. */
  PyObject* setLimit(int type);
  /** Returns the type of constraint. */
  PyObject* getLimit();
};

#endif
```

```
/***************************************************************************
                          soundactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef SOUNDACTUATOR_H
#define SOUNDACTUATOR_H

#include <actuator.h>

/**A SoundActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">SoundActuator : public Actuator  {
public:
        SoundActuator();
        ~SoundActuator();
  /** Filename of the sound file. */
  PyObject* setFilename(char* name);
  /** Get filename of the sound file. */
  PyObject* getFilename();
};

#endif
```

```
/***************************************************************************
                          propertyactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef PROPERTYACTUATOR_H
#define PROPERTYACTUATOR_H

#include <actuator.h>

/**A PropertyActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">PropertyActuator : public Actuator  {
public:
        PropertyActuator();
        ~PropertyActuator();
  /** Set the property on which to operate. If there is no property of this name, the
call is ignored. */
  PyObject* setProperty(char* name);
  /** Return the property on which the actuator operates. */
  PyObject* getProperty();
  /** Set the value with which the actuator operates. If the value is not compatible
with the type of the property, the subsequent action is ignored. */
  PyObject* setValue(char* value);
  /** Returns the value with which the actuator operates. */
  PyObject* getValue();
};

#endif
```

```
/***************************************************************************
                          sceneactuator.h  -  description
                             ------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef SCENEACTUATOR_H
#define SCENEACTUATOR_H

#include <actuator.h>

/**A SceneActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">SceneActuator : public Actuator  {
public:
        SceneActuator();
        ~SceneActuator();
  /** Set flag to 1 to restart the scene. */
  PyObject* setUseRestart(int flag);
  /** Return whether the scene will be restarted. */
  PyObject* getUseRestart();
  /** Set the scene to switch to. */
  PyObject* setScene(char* sceneName);
  /** Return the scene to switch to. */
  PyObject* getScene();
  /** Set the camera to switch to. */
  PyObject* setCamera(char* camera);
  /** Return the camera to switch to. */
  PyObject* getCamera();
};

#endif
```

```
/***************************************************************************
                          cameraactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef CAMERAACTUATOR_H
#define CAMERAACTUATOR_H

#include <actuator.h>

/**A CameraActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">CameraActuator : public Actuator  {
public:
        CameraActuator();
        ~CameraActuator();
};

#endif
```

```
/***************************************************************************
                          addobjectactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ADDOBJECTACTUATOR_H
#define ADDOBJECTACTUATOR_H

#include <actuator.h>

/**A AddObjectActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">AddObjectActuator : public Actuator  {
public:
        AddObjectActuator();
        ~AddObjectActuator();
  /** Sets the object that will be added. There has to be an object of this name. If
not, this function does nothing. */
  PyObject* setObject(char* name);
  /** Returns the name of the object that will be added. */
  PyObject* getObject();
  /** Sets the lifetime of the object that will be added, in frames. If the duration
is negative, it is set to 0. */
  PyObject* setTime(int duration);
  /** Returns the lifetime of the object that will be added. */
  PyObject* getTime();
  /** Assign this velocity to the created object. */
  PyObject* setLinearVelocity(float vx, float vy, float vz);
  /** Returns the linear velocity that will be assigned to the created object. */
  PyObject* getLinearVelocity();
};

#endif
```

```
/***************************************************************************
                          groupactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef GROUPACTUATOR_H
#define GROUPACTUATOR_H

#include <actuator.h>

/**A GroupActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">GroupActuator : public Actuator  {
public:
        GroupActuator();
        ~GroupActuator();
};

#endif
```

```
/***************************************************************************
                          objectactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef OBJECTACTUATOR_H
#define OBJECTACTUATOR_H

#include <actuator.h>

/**A ObjectActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">ObjectActuator : public Actuator  {
public:
        ObjectActuator();
        ~ObjectActuator();
  /** bla */
  PyObject* getForce();
        /** bla */
        PyObject* setForce(float x, float y, float z, int toggle);
  /** bla */
  PyObject* getTorque();
        /** bla */
        PyObject* setTorque(float x, float y, float z, int toggle);
  /** bla */
  PyObject* getDLoc();
        /** bla */
        PyObject* setDLoc(float x, float y, float z, int toggle);
  /** bla */
  PyObject* getDRot();
        /** bla */
        PyObject* setDRot(float x, float y, float z, int toggle);
  /** bla */
  PyObject* getLinearVelocity();
        /** bla */
        PyObject* setLinearVelocity(float x, float y, float z, int toggle);
```

```
  /** bla */
  PyObject* getAngularVelocity();
        /** bla */
        PyObject* setAngularVelocity(float x, float y, float z, int toggle);
};

#endif
```

```
/***************************************************************************
                          ipoactuator.h  -  description
                             -------------------
    begin                : Wed Dec 20 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef IPOACTUATOR_H
#define IPOACTUATOR_H

#include <actuator.h>

/**A IpoActuator is a special kind of a general actuator.
  *@author Jan Walter
  */

class Actuator.html">IpoActuator : public Actuator  {
public:
        IpoActuator();
        ~IpoActuator();
  /** Possible values for mode "PLAY", "PINGPONG", "FLIPPER", "LOOPSTOP", "LOOPEND".
*/
  PyObject* set(char* mode, int startFrame, int stopFrame, char* forceToggle);
  /** bla */
  PyObject* setKey2Key(char* previous, char* cycle, char* hold, char* propertyName);
  /** bla */
  PyObject* setProperty(char* propertyName);
};

#endif
```

```
/***************************************************************************
                          object.h  -  description
                             ------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef OBJECT_H
#define OBJECT_H

struct PyObject;

/**Baseclass for all game objects.
  *@author Jan Walter
  */

class Object {
public:
        Object();
        ~Object();
  /** A list of lists is used to set the orientation matrix within Blender's game
engine. */
  PyObject* setOrientation(PyObject* pylist);
  /** A list of lists is returned for the orientation matrix within Blender's game
engine. */
  PyObject* getOrientation();
};

#endif
```

```
/***************************************************************************
                          gameobject.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef GAMEOBJECT_H
#define GAMEOBJECT_H

#include <object.h>

/**A GameObject is a special kind of a general object in Blender's game engine.
  *@author Jan Walter
  */

class Object.html">GameObject : public Object  {
public:
        GameObject();
        ~GameObject();
  /** bla */
  PyObject* setVisible(int visible);
  /** Takes a Python list of three elements (x, y, z). */
  PyObject* setPosition(PyObject* pylist);
  /** Returns a Python list of three elements (x, y, z). */
  PyObject* getPosition();
  /** Returns a Python list of three elements. */
  PyObject* getLinearVelocity();
  /** Returns a Python list of three elements. */
  PyObject* getVelocity();
  /** bla */
  PyObject* getMass();
  /** Returns a Python list of three elements. */
  PyObject* getReactionForce();
  /** bla */
  PyObject* restoreDynamics();
  /** bla */
  PyObject* suspendDynamics();
};

#endif
```

Documentation generated by jan@nvidea on Thu Dec 21 14:04:43 CET 2000

```
/***************************************************************************
                          camera.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef CAMERA_H
#define CAMERA_H

#include <gameobject.h>

/**A Camera is a special kind of a GameObject in Blender's game engine.
  *@author Jan Walter
  */

class Camera : public GameObject  {
public:
        Camera();
        ~Camera();
};

#endif
```

```
/***************************************************************************
                          lightobject.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef LIGHTOBJECT_H
#define LIGHTOBJECT_H

#include <gameobject.h>

/**A LightObject is a special kind of a GameObject in Blender's game engine.
  *@author Jan Walter
  */

class LightObject : public GameObject  {
public:
        LightObject();
        ~LightObject();
};

#endif
```

```
/***************************************************************************
                          emptyobject.h  -  description
                             ------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef EMPTYOBJECT_H
#define EMPTYOBJECT_H

#include <gameobject.h>

/**A EmptyObject is a special kind of a GameObject in Blender's game engine.
  *@author Jan Walter
  */

class EmptyObject : public GameObject  {
public:
        EmptyObject();
        ~EmptyObject();
};

#endif
```

```
/***************************************************************************
                          controller.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef CONTROLLER_H
#define CONTROLLER_H

#include <logicbrick.h>

/**Baseclass for all controllers.
  *@author Jan Walter
  */

class Controller : public LogicBrick  {
public:
        Controller();
        ~Controller();
};

#endif
```

```
/***************************************************************************
                          orcontroller.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ORCONTROLLER_H
#define ORCONTROLLER_H

#include <controller.h>

/**A ORController is a special kind of a general controller.
  *@author Jan Walter
  */

class Controller.html">ORController : public Controller  {
public:
        ORController();
        ~ORController();
};

#endif
```

```
/***************************************************************************
                          andcontroller.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef ANDCONTROLLER_H
#define ANDCONTROLLER_H

#include <controller.h>

/**A ANDController is a special kind of a general controller.
  *@author Jan Walter
  */

class Controller.html">ANDController : public Controller  {
public:
        ANDController();
        ~ANDController();
};

#endif
```

```
/***************************************************************************
                          logicbrick.h  -  description
                             -------------------
    begin                : Thu Dec 21 2000
    copyright            : (C) 2000 by Jan Walter
    email                : jan@blender.nl
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef LOGICBRICK_H
#define LOGICBRICK_H

struct PyObject;

/**Baseclass for all sensors, controller, and actuators.
  *@author Jan Walter
  */

class LogicBrick {
public:
        LogicBrick();
        ~LogicBrick();
  /** Returns the owner of the Python script. */
  PyObject* getOwner();
};

#endif
```
