
Blender-VR Temple Source

Release 1.0

Blender-VR Team

September 14, 2015

1	Source Code	2
1.1	src package	2
2	Indices and tables	10
	Python Module Index	11

The documentation of the Blender-VR Temple demo project

1.1 src package

1.1.1 Submodules

Artificial Intelligence

Control the enemies behaviour (bats, ghosts, pendulum)

```
src.ai.BL_EnemyArmatureObject  
    alias of BL_ArmatureObject
```

```
class src.ai.Base (parent)  
    Bases: src.base.Base  
  
    bats  
  
    ghosts  
  
    loop ()  
  
    pendulums  
  
    spawnEnemies ()  
        populate enemies  
  
    trailSeek ()  
        see if we are to be activated  
  
    trailSeeker (ob, controller, actuator)  
        Store the object we use to evaluate the trail for enemy spawning
```

```
class src.ai.Bat (scene, obj, target, speed, events, logger)  
    Bases: src.ai.FlyingEnemy
```

```
    enemy = 'BAT'  
    ray_filter = 'bat'
```

```
class src.ai.Enemy (speed, events, logger)  
    Bases: builtins.object  
  
    addObject (scene, object_name, object_origin)  
        Spawn a new object in the game
```

attack (*origin*)

Check if enemy is close enough to eat

attack_distance_squared = 0.01

classmethod calculateId ()

Generates a new id based on the number of added instances

changeState ()

Called when the object changes to a relevant state, called from Logic Bricks

end ()

End the object, called from Logic Bricks

evade_distance_squared = 100.0

static getState (*state*)

Return the bitwise flag corresponding to this state

classmethod hit (*camera, origin, direction*)

Try to hit an enemy from this origin at this direction If succeeds, send a message to end the object

init ()

Initialize the object, called from Logic Bricks

instances = 0

kill ()

Send message to eliminate the object It is called when we hit the enemy or the enemy hits us

ray_filter = ''

setSound (*sound*)

Setup OSC sound engine, called from sound.py

sound_source

Return the object to use as reference for the sound origin

subject

Message subject to use with the Message sensor to end the object

class `src.ai.FlyingEnemy` (*name, scene, obj, target, speed, events, logger*)

Bases: `src.ai.Enemy`

class `src.ai.Ghost` (*scene, obj, target, speed, events, logger*)

Bases: `src.ai.FlyingEnemy`

activation_distance = 30.0

attack_distance_squared = 0.25

enemy = 'GHOST'

ray_filter = 'ghost'

`src.ai.KX_EnemyGameObject`

alias of `KX_GameObject`

class `src.ai.Pendulum` (*scene, obj, speed, events, logger*)

Bases: `src.ai.Enemy`

attack()
Called from Logic Brick callback We are already hitting the player

end()
End the object, called from Logic Bricks

enemy = 'PENDULUM'

evade(*origin*)
Check if enemy is too far

ray_filter = 'pendulum'

class `src.ai.Seeker` (*ob, controller, actuator*)

Bases: `builtins.object`

getOrientation (*frame, callback, user_data=None*)
Get orientation matrix of the animation at a given frame

Parameters

- **frame** (*int*) – animation frame
- **callback** (*function(mathutils.Matrix, user_data)*) – callback function
- **user_data** (*Object*) – user data passed back to callback function

getPosition (*frame, callback, user_data=None*)
Get position of the animation at a given frame

Parameters

- **frame** (*int*) – animation frame
- **callback** (*function(mathutils.Vector, user_data)*) – callback function
- **user_data** (*Object*) – user data passed back to callback function

getTransform (*frame, callback, user_data=None*)
Get position and orientation matrix of the animation at a given frame

Parameters

- **frame** (*int*) – animation frame
- **callback** (*function(((mathutils.Vector, mathutils.Matrix), user_data)*) – callback function
- **user_data** (*Object*) – user data passed back to callback function

loop()
activate the actuator to return any stacked query

`src.ai.attacked` (*cont*)

Called from Logic Bricks upon collision with enemy

Only pendulum collides this way, the other objects Use the steering actuator which is incompatible with physics sensors

`src.ai.changeState` (*cont*)

Called from Logic Bricks upon change to relevant states (e.g., start chasing, or end object)

`src.ai.trailSeeking` (*cont*)

activate trail seeking actuator

Base

Basic class for all inherited classes.

```
class src.base.Base (parent)
    Bases: builtins.object

    loop ()
```

```
class src.base.Pipe (parent, name)
    Bases: builtins.object

    Pipe commands from the base class (children) to the events module

    e.g., in io.py you can do: self.setFlashlightMode(power=True) and this will call events.setFlashlightMode(power=True)
```

Debug

This module is intended to run and debug the demo without BlenderVR

```
class src.debug.Base (parent)
    Bases: src.base.Base

    loop ()
        Run once per frame called from a callback
```

Events Manager

Centralize the communication between the game elements, the navigation devices, the input systems and the sound engine.

```
class src.events.Base (parent)
    Bases: src.base.Base

    evadeEnemy (enemy)
        Enemy got too distant, it can go away

    gameOver ()
        The game time is over

    hitByEnemy (enemy)
        Enemy got the upper hand

    hitEnemy (enemy)
        Enemy got hit, congratulations

    setFlashlightMode (power=True)
        Activate or Deactivate the flashlight

    setSonarMode (power=True)
        Activate or Deactivate the sonar

    spawnEnemy (enemy)
        A new enemy got spawned

    startLap ()
        Start a new lap

    throwRock ()
        Throw a rock (to hit a pendulum)
```

Input/Output

Takes the three different inputs, process the data and call the corresponding event. It also handles head transformation/navigation.

```

class src.io.Base (parent)
    Bases: src.base.Base

    enableHeadTrack (user)
        Use Headtrack (instead of mouse) to control the scene Called from BlenderVR processor file

        Parameters user – BlenderVR User

    flashlightButton ()
        Flashlight button was pressed

    head_direction
        Get the direction of the player's head

        Return type mathutils.Vector (normalized)

    head_orientation
        Get the direction of the player's head

        Return type mathutils.Quaternion

    head_position
        Get the direction of the player's head

        Return type mathutils.Vector

    is_flashlight

    is_sonar

    loop ()

    rockButton ()
        Rock button was pressed

    sonarButton ()
        Sonar button was pressed

```

Logger

Handles all printing/logging operations

```

class src.logger.Base
    Bases: builtins.object

class src.logger.Logger
    Bases: builtins.object

class src.logger.Print (function)
    Bases: builtins.object

class src.logger.Printer
    Bases: builtins.object

```

Scoring System

Keep track of player's progress, score and final results

```
class src.score.Base (parent)
    Bases: src.base.Base

    evade (enemy)
        Enemy goes away

    hit (enemy)
        Hit (kill) an enemy

    hitBy (enemy)
        Got attacked by an enemy

    spawn (enemy)
        Add a new enemy in the game
```

Sound

Connected with the game events systems it deals directly with the OSC server It also provides a phantom layer, which mimics OSC with local BGE sound resources.

```
class src.sound.AudaspacesoundEngine (logger)
    Bases: src.sound.SoundEngine

class src.sound.AudaspacesoundObject (engine, kx_object)
    Bases: builtins.object

    play (sound, loop=False, volume=0.5)
        Load and play sound file

class src.sound.Base (parent)
    Bases: src.base.Base

    setOSCUser (user)
        Specify the OSC User to use

    setVolumeHigh ()
        High volume, sonar is on

    setVolumeLow ()
        Low volume, flashlight is on

    setVolumeNormal ()
        Normal volume, initial

class src.sound.Bat (engine, sound_source, force_fallback=False)
    Bases: src.sound.Enemy

    sound_end = 'bat_end.wav'

    sound_init = 'bat.wav'

class src.sound.Enemy (engine, sound_source, sound_init, sound_end, force_fallback)
    Bases: builtins.object

    audio_folder = '../audio/'

    osc = None
```

playEnd()
Play sound for when the object ends (e.g., is hit by rock)

playInit()
Play sound for when the object is active (e.g., flying)

sound_end

sound_init

class `src.sound.Ghost` (*engine, sound_source, force_fallback=False*)
Bases: `src.sound.Enemy`

sound_end = 'ghost_end.wav'

sound_init = 'ghost.wav'

class `src.sound.OSCSoundEngine` (*osc, logger*)
Bases: `src.sound.SoundEngine`

getUser()
called from the OSC objects

setUser (*user*)
called from the processor file

class `src.sound.OSCSoundObject` (*osc, engine, kx_object*)
Bases: `builtins.object`

play (*sound, loop=False, volume=0.5*)
Load and play sound file

Parameters **sound** (*str*) – The sound file filepath

class `src.sound.Pendulum` (*engine, sound_source, force_fallback=False*)
Bases: `src.sound.Enemy`

sound_end = 'pendulum_end.wav'

sound_init = 'pendulum.wav'

class `src.sound.SoundEngine` (*logger, volumes*)
Bases: `builtins.object`

setVolumeHigh()
Set the highest volume level

setVolumeLow()
Set the lowest volume level

setVolumeNormal()
Set the regular (initial) volume level

Timeline

Control the flux of the game, and the time related events (spawn of enemies, game end, game start, ...).

class `src.timeline.Base` (*parent*)
Bases: `src.base.Base`

loop()

1.1.2 Module contents

Blender-VR Temple Source

```
class src.Temple
    Bases: builtins.object
    ai
    bumpSpeed()
        Increase global speed
    debug
    events
    io
    is_debug
    logger
    run()
        Run once per frame, called from the processor file
    score
    sound
    speed
    timeline
src.main()
```

Indices and tables

- *genindex*
- *modindex*
- *search*

S

src, 9
src.ai, 2
src.base, 4
src.debug, 5
src.events, 5
src.io, 5
src.logger, 6
src.score, 6
src.sound, 7
src.timeline, 8

A

activation_distance (src.ai.Ghost attribute), 3
addObject() (src.ai.Enemy method), 2
ai (src.Temple attribute), 9
attack() (src.ai.Enemy method), 2
attack() (src.ai.Pendulum method), 3
attack_distance_squared (src.ai.Enemy attribute), 3
attack_distance_squared (src.ai.Ghost attribute), 3
attacked() (in module src.ai), 4
AudaspaceSoundEngine (class in src.sound), 7
AudaspaceSoundObject (class in src.sound), 7
audio_folder (src.sound.Enemy attribute), 7

B

Base (class in src.ai), 2
Base (class in src.base), 5
Base (class in src.debug), 5
Base (class in src.events), 5
Base (class in src.io), 6
Base (class in src.logger), 6
Base (class in src.score), 7
Base (class in src.sound), 7
Base (class in src.timeline), 8
Bat (class in src.ai), 2
Bat (class in src.sound), 7
bats (src.ai.Base attribute), 2
BL_EnemyArmatureObject (in module src.ai), 2
bumpSpeed() (src.Temple method), 9

C

calculateId() (src.ai.Enemy class method), 3
changeState() (in module src.ai), 4
changeState() (src.ai.Enemy method), 3

D

debug (src.Temple attribute), 9

E

enableHeadTrack() (src.io.Base method), 6
end() (src.ai.Enemy method), 3

end() (src.ai.Pendulum method), 4
Enemy (class in src.ai), 2
Enemy (class in src.sound), 7
enemy (src.ai.Bat attribute), 2
enemy (src.ai.Ghost attribute), 3
enemy (src.ai.Pendulum attribute), 4
evade() (src.ai.Pendulum method), 4
evade() (src.score.Base method), 7
evade_distance_squared (src.ai.Enemy attribute), 3
evadeEnemy() (src.events.Base method), 5
events (src.Temple attribute), 9

F

flashlightButton() (src.io.Base method), 6
FlyingEnemy (class in src.ai), 3

G

gameOver() (src.events.Base method), 5
getOrientation() (src.ai.Seeker method), 4
getPosition() (src.ai.Seeker method), 4
getState() (src.ai.Enemy static method), 3
getTransform() (src.ai.Seeker method), 4
getUser() (src.sound.OSCSoundEngine method), 8
Ghost (class in src.ai), 3
Ghost (class in src.sound), 8
ghosts (src.ai.Base attribute), 2

H

head_direction (src.io.Base attribute), 6
head_orientation (src.io.Base attribute), 6
head_position (src.io.Base attribute), 6
hit() (src.ai.Enemy class method), 3
hit() (src.score.Base method), 7
hitBy() (src.score.Base method), 7
hitByEnemy() (src.events.Base method), 5
hitEnemy() (src.events.Base method), 5

I

init() (src.ai.Enemy method), 3
instances (src.ai.Enemy attribute), 3

io (src.Temple attribute), 9
 is_debug (src.Temple attribute), 9
 is_flashlight (src.io.Base attribute), 6
 is_sonar (src.io.Base attribute), 6

K

kill() (src.ai.Enemy method), 3
 KX_EnemyGameObject (in module src.ai), 3

L

Logger (class in src.logger), 6
 logger (src.Temple attribute), 9
 loop() (src.ai.Base method), 2
 loop() (src.ai.Seeker method), 4
 loop() (src.base.Base method), 5
 loop() (src.debug.Base method), 5
 loop() (src.io.Base method), 6
 loop() (src.timeline.Base method), 8

M

main() (in module src), 9

O

osc (src.sound.Enemy attribute), 7
 OSCSoundEngine (class in src.sound), 8
 OSCSoundObject (class in src.sound), 8

P

Pendulum (class in src.ai), 3
 Pendulum (class in src.sound), 8
 pendulums (src.ai.Base attribute), 2
 Pipe (class in src.base), 5
 play() (src.sound.AudaspaceSoundObject method), 7
 play() (src.sound.OSCSoundObject method), 8
 playEnd() (src.sound.Enemy method), 7
 playInit() (src.sound.Enemy method), 8
 Print (class in src.logger), 6
 Printer (class in src.logger), 6

R

ray_filter (src.ai.Bat attribute), 2
 ray_filter (src.ai.Enemy attribute), 3
 ray_filter (src.ai.Ghost attribute), 3
 ray_filter (src.ai.Pendulum attribute), 4
 rockButton() (src.io.Base method), 6
 run() (src.Temple method), 9

S

score (src.Temple attribute), 9
 Seeker (class in src.ai), 4
 setFlashlightMode() (src.events.Base method), 5
 setOSCUser() (src.sound.Base method), 7
 setSonarMode() (src.events.Base method), 5

setSound() (src.ai.Enemy method), 3
 setUser() (src.sound.OSCSoundEngine method), 8
 setVolumeHigh() (src.sound.Base method), 7
 setVolumeHigh() (src.sound.SoundEngine method), 8
 setVolumeLow() (src.sound.Base method), 7
 setVolumeLow() (src.sound.SoundEngine method), 8
 setVolumeNormal() (src.sound.Base method), 7
 setVolumeNormal() (src.sound.SoundEngine method), 8
 sonarButton() (src.io.Base method), 6
 sound (src.Temple attribute), 9
 sound_end (src.sound.Bat attribute), 7
 sound_end (src.sound.Enemy attribute), 8
 sound_end (src.sound.Ghost attribute), 8
 sound_end (src.sound.Pendulum attribute), 8
 sound_init (src.sound.Bat attribute), 7
 sound_init (src.sound.Enemy attribute), 8
 sound_init (src.sound.Ghost attribute), 8
 sound_init (src.sound.Pendulum attribute), 8
 sound_source (src.ai.Enemy attribute), 3
 SoundEngine (class in src.sound), 8
 spawn() (src.score.Base method), 7
 spawnEnemies() (src.ai.Base method), 2
 spawnEnemy() (src.events.Base method), 5
 speed (src.Temple attribute), 9
 src (module), 9
 src.ai (module), 2
 src.base (module), 4
 src.debug (module), 5
 src.events (module), 5
 src.io (module), 5
 src.logger (module), 6
 src.score (module), 6
 src.sound (module), 7
 src.timeline (module), 8
 startLap() (src.events.Base method), 5
 subject (src.ai.Enemy attribute), 3

T

Temple (class in src), 9
 throwRock() (src.events.Base method), 5
 timeline (src.Temple attribute), 9
 trailSeek() (src.ai.Base method), 2
 trailSeeker() (src.ai.Base method), 2
 trailSeeking() (in module src.ai), 4