cocos.actions.interval_actions module

Interval Action

Interval Actions

An interval action is an action that takes place within a certain period of time. It has an start time, and a finish time. The finish time is the parameter `duration` plus the start time.

These *IntervalAction* have some interesting properties, like:

- They can run normally (default)
- They can run reversed with the *Reverse* action.
- They can run with the time altered with the *Accelerate*, *AccelDeccel* and *Speed* actions.

For example, you can simulate a Ping Pong effect running the action normally and then running it again in Reverse mode.

Example:

```
ping_pong_action = action + Reverse( action )
```

Available IntervalActions

- *MoveTo*
- *MoveBy*
- *JumpTo*
- *JumpBy*
- *Bezier*
- *Blink*
- *RotateTo*
- *RotateBy*
- *ScaleTo*
- *ScaleBy*
- *FadeOut*
- *FadeIn*
- *FadeTo*
- *Delay*
- *RandomDelay*

Modifier actions

- *Accelerate*
- *AccelDeccel*
- *Speed*

Examples:

```
move = MoveBy( (200,0), duration=5 )  # Moves 200 pixels to the right in 5 seconds.

move = MoveTo( (320,240), duration=5) # Moves to the pixel (320,240) in 5 seconds

jump = JumpBy( (320,0), 100, 5, duration=5) # Jumps to the right 320 pixels
                                            # doing 5 jumps of 100 pixels
                                            # of height in 5 seconds

accel_move = Accelerate(move)               # accelerates action move
```

class **Lerp**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Interpolate between values for some specified attribute

init(*attrib*, *start*, *end*, *duration*)

Init method.

Parameters:

attrib : *string*

The name of the attrbiute where the value is stored

start : *float*

The start value

end : *float*

The end value

duration : *float*

Duration time in seconds

update(*t*)

class **MoveTo**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Moves a *CocosNode* object to the position x,y. x and y are absolute coordinates by modifying it's position attribute.

Example:

```
# Move the sprite to coords x=50, y=10 in 8 seconds

action = MoveTo( (50,10), 8 )
sprite.do( action )
```

init(*dst_coords*, *duration=5*)

Init method.

Parameters:

dst_coords : *(x,y)*

Coordinates where the sprite will be placed at the end of the action

duration : *float*

Duration time in seconds

start()

update(*t*)

class **MoveBy**(**args*, ***kwargs*)

Bases: **cocos.actions.interval_actions.MoveTo**

Moves a *CocosNode* object x,y pixels by modifying it's position attribute. x and y are relative to the position of the object. Duration is is seconds.

Example:

```
# Move the sprite 50 pixels to the left in 8 seconds
action = MoveBy( (-50,0), 8 )
sprite.do( action )
```

init(*delta*, *duration=5*)

Init method.

Parameters:

delta : *(x,y)*

Delta coordinates

duration : *float*

Duration time in seconds

start()

class **Jump**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Moves a *CocosNode* object simulating a jump movement by modifying it's position attribute.

Example:

```
action = Jump(50,200, 5, 6)     # Move the sprite 200 pixels to the right
sprite.do( action )             # in 6 seconds, doing 5 jumps
                                # of 50 pixels of height
```

init(*y=150*, *x=120*, *jumps=1*, *duration=5*)

Init method

Parameters:

y : *integer*

Height of jumps

x : *integer*
: horizontal movement relative to the startin position

jumps : *integer*
: quantity of jumps

duration : *float*
: Duration time in seconds

start()

update(*t*)

class **JumpTo**(**args*, ***kwargs*)

Bases: **cocos.actions.interval_actions.JumpBy**

Moves a *CocosNode* object to a position simulating a jump movement by modifying it's position attribute.

Example:

```
action = JumpTo(50,200, 5, 6)  # Move the sprite 200 pixels to the right
sprite.do( action )            # in 6 seconds, doing 5 jumps
                               # of 50 pixels of height
```

start()

class **JumpBy**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Moves a *CocosNode* object simulating a jump movement by modifying it's position attribute.

Example:

```
# Move the sprite 200 pixels to the right and up
action = JumpBy((100,100),200, 5, 6)
sprite.do( action )            # in 6 seconds, doing 5 jumps
                               # of 200 pixels of height
```

init(*position=(0, 0)*, *height=100*, *jumps=1*, *duration=5*)

Init method

Parameters:

position : *integer x integer tuple*
: horizontal and vertical movement relative to the starting position

height : *integer*
: Height of jumps

jumps : *integer*
quantity of jumps

duration : *float*
Duration time in seconds

start()

update(*t*)

class **Bezier**(**args*, ***kwargs*)

Bases: `cocos.actions.base_actions.IntervalAction`

Moves a *CocosNode* object through a bezier path by modifying it's position attribute.

Example:

```
action = Bezier( bezier_conf.path1, 5 )    # Moves the sprite using the
sprite.do( action )                        # bezier path 'bezier_conf.path1'
                                           # in 5 seconds
```

init(*bezier*, *duration=5*, *forward=True*)

Init method

Parameters:

bezier : *bezier_configuration instance*
A bezier configuration

duration : *float*
Duration time in seconds

start()

update(*t*)

Rotate

alias of `RotateBy`

class **RotateTo**(**args*, ***kwargs*)

Bases: `cocos.actions.base_actions.IntervalAction`

Rotates a *CocosNode* object to a certain angle by modifying it's rotation attribute. The direction will be decided by the shortest angle.

Example:

```
# rotates the sprite to angle 180 in 2 seconds
action = RotateTo( 180, 2 )
sprite.do( action )
```

init(*angle*, *duration*)

Init method.

Parameters:

angle : *float*

Destination angle in degrees.

duration : *float*

Duration time in seconds

start()

update(*t*)

class **RotateBy**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Rotates a *CocosNode* object clockwise a number of degrees by modiying it's rotation attribute.

Example:

```
# rotates the sprite 180 degrees in 2 seconds
action = RotateBy( 180, 2 )
sprite.do( action )
```

init(*angle*, *duration*)

Init method.

Parameters:

angle : *float*

Degrees that the sprite will be rotated. Positive degrees rotates the sprite clockwise.

duration : *float*

Duration time in seconds

start()

update(*t*)

class **ScaleTo**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Scales a *CocosNode* object to a zoom factor by modifying it's scale attribute.

Example:

```
# scales the sprite to 5x in 2 seconds
action = ScaleTo( 5, 2 )
sprite.do( action )
```

init(*scale*, *duration=5*)

Init method.

Parameters:

scale : *float*

scale factor

duration : *float*

Duration time in seconds

start()

update(*t*)

class **ScaleBy**(**args*, ***kwargs*)

Bases: **cocos.actions.interval_actions.ScaleTo**

Scales a *CocosNode* object a zoom factor by modifying it's scale attribute.

Example:

```
# scales the sprite by 5x in 2 seconds
action = ScaleBy( 5, 2 )
sprite.do( action )
```

start()

class **Delay**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Delays the action a certain amount of seconds

Example:

```
action = Delay(2.5)
sprite.do( action )
```

init(*delay*)

Init method

Parameters:

delay : *float*

Seconds of delay

class **RandomDelay**(**args*, ***kwargs*)

Bases: **cocos.actions.interval_actions.Delay**

Delays the actions between *min* and *max* seconds

Example:

```
action = RandomDelay(2.5, 4.5)      # delays the action between 2.5 and 4.5 seco
sprite.do( action )
```

init(*low*, *hi*)

Init method

Parameters:

low : *float*

Minimun seconds of delay

hi : *float*

Maximun seconds of delay

class **FadeOut**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Fades out a *CocosNode* object by modifying it's opacity attribute.

Example:

```
action = FadeOut( 2 )
sprite.do( action )
```

init(*duration*)

Init method.

Parameters: *duration* : *float*

Seconds that it will take to fade

update(*t*)

class **FadeIn**(**args*, ***kwargs*)

Bases: **cocos.actions.interval_actions.FadeOut**

Fades in a *CocosNode* object by modifying it's opacity attribute.

Example:

```
action = FadeIn( 2 )
sprite.do( action )
```

update(*t*)

class **FadeTo**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Fades a *CocosNode* object to a specific alpha value by modifying it's opacity attribute.

Example:

```
action = FadeTo( 128, 2 )
sprite.do( action )
```

init(*alpha*, *duration*)

Init method.

Parameters:

alpha : *float*

0-255 value of opacity

duration : *float*

Seconds that it will take to fade

start()

update(*t*)

class **Blink**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Blinks a *CocosNode* object by modifying it's visible attribute

The action ends with the same visible state than at the start time.

Example:

```
# Blinks 10 times in 2 seconds
action = Blink( 10, 2 )
sprite.do( action )
```

init(*times*, *duration*)

Init method.

Parameters:

times : *integer*

Number of times to blink

duration : *float*

Duration time in seconds

start()

update(*t*)

class **Accelerate**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Changes the acceleration of an action

Example:

```
# rotates the sprite 180 degrees in 2 seconds clockwise
# it starts slow and ends fast
action = Accelerate( Rotate( 180, 2 ), 4 )
sprite.do( action )
```

init(*other*, *rate=2*)

Init method.

Parameters:

other : *IntervalAction*

The action that will be affected

rate : *float*

The acceleration rate. 1 is linear. the new t is t**rate

start()

update(*t*)

class **AccelDeccel**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Makes an action change the travel speed but retain near normal speed at the beginning and ending.

Example:

```
# rotates the sprite 180 degrees in 2 seconds clockwise
# it starts slow, gets fast and ends slow
action = AccelDeccel( RotateBy( 180, 2 ) )
sprite.do( action )
```

init(*other*)

Init method.

Parameters:

other : *IntervalAction*

The action that will be affected

start()

update(*t*)

class **Speed**(**args*, ***kwargs*)

Bases: **cocos.actions.base_actions.IntervalAction**

Changes the speed of an action, making it take longer (speed>1) or less (speed<1)

Example:

```
# rotates the sprite 180 degrees in 1 secondclockwise
action = Speed( Rotate( 180, 2 ), 2 )
sprite.do( action )
```

init(*other*, *speed*)

Init method.

Parameters:

other : *IntervalAction*

The action that will be affected

speed : *float*

The speed change. 1 is no change. 2 means twice as fast, takes half the time 0.5 means half as fast, takes double the time

start()

update(*t*)