
KBEngine?

KBEngine(Unity3D, OGRE, Cocos2d-x, HTML5,)
Python()

<http://www.kbengine.org> Releases sources :

<https://github.com/kbengine/kbengine/releases/latest> binarys :

<https://sourceforge.net/projects/kbengine/files/> Demo sources unity3d :

https://github.com/kbengine/kbengine_unity3d_demo/releases/latest unity3d

: https://github.com/kbengine/kbengine_unity3d_warring/releases/latest

ogre : https://github.com/kbengine/kbengine_ogre_demo/releases/latest

html5 : https://github.com/kbengine/kbengine_html5_demo/releases/latest

Docs docs : <http://www.kbengine.org/docs/> API :

<https://github.com/kbengine/kbengine/tree/master/docs> Support Email :

kbesrv@gmail.com Maillist :

https://groups.google.com/d/forum/kbengine_maillist



KBEngine , [GitHub](#)

?

forking kbengine_
issue

KBEngine

Mailbox:

```
(allClientsotherClientsclientEntity)  
MailboxC++IDMailbox
```

```
: Mailboxdef
```

```
client ()client(Mailbox)  
baseBaseappBaseappbase(Mailbox)  
cellCellappCellappcell(Mailbox)
```

```
:
```

```
Avatar.defclient:
```

```
<ClientMethods>  
    <hello>  
    </hello>  
</ClientMethods>
```

```
client\Avatar.py
```

```
class Avatar:  
    def hello(self):  
        print("hello")
```

```
GUIConsoleDebug()  
Baseapp(Avatar)ID, ID(Avatar)Mailbox:  
>>> KBEngine.entities[ID].client.hello()
```

log"hello",

KBE_ROOT:

KBEngineKBEngine

KBE_RES_PATH:

KBEngineKBEngine

KBE_HYBRID_PATH:

KBEngineKBEngine

entities.xml:

kbengine_defaults.xml:

KBEngine [cellappbaseapp](#)loginapp
[kbengine.xml](#)xml

kbengine.xml:

[cellappbaseapp](#)loginapp

kbengine_defaults.xml

Pythonobject

?

<http://www.kbengine.org/docs/programming/entity>

entity

:

View

ViewViewView

ViewView

MMORPG

:

Witness

Witness **cell**Viewwitness **cell**cellappViewWitness

NPCOnWitnessCPU

Space

KBEnginecellappView

...

: **Space**

cell

cell

Base.cell **CellMailbox**

cellappcell

cellappNcellcell

base

baseappBaseBaseBaseMailbox, **Entity.base**

client

Mailbox, **Base.client**

cellapp

CellappViewAI

: **cellapp**

baseapp

Baseapp()

: **baseapp**

real

cellcellcell

ghost

cellappspaceNcell
spacespacecellcell
(CELL_PUBLICcell)

ghost **real**

vector3

3D
xyz

```
import Math v = Math.Vector3()
```




[Name]	[Bytes]	
UINT8	1	
UINT16	2	
UINT32	4	
UINT64	8	
INT8	1	
INT16	2	
INT32	4	
INT64	8	
FLOAT	4	
DOUBLE	8	
VECTOR2	12	
VECTOR3	16	
VECTOR4	20	
STRING	N	
UNICODE	N	
PYTHON	N	
PY_DICT	N	
PY_TUPLE		N
PY_LIST	N	
MAILBOX	N	
BLOB	N	

KBEngine

KBEngine, space




Entity



```
def login( username, password ):
def createAccount( username, password ):
def reloginBaseapp():
def player( ):
def resetPassword(username):
def bindAccountEmail( emailaddress ):
def newPassword( oldpassword, newpassword ):
def findEntity( entityID ):
def getSpaceData( key ):
```



component	string
entities	Entities
entity_uuid	uint64
entity_id	int32
spaceID	int32

```
def login( username, password ):
```

```
KBEngine  
UIUI"login"usernamepassword
```

```
    username string  
    password string
```

```
def createAccount( username, password ):
```

```
KBEngine  
UIUI"createAccount"usernamepassword
```

```
    username string  
    password string
```

```
def reloginBaseapp( ):
```

```
KBEngine()  
UIUI"reloginBaseapp"
```

```
def player( ):
```

Entity, ()

```
def resetPassword( username ):
```

loginapp, ()

username string

```
def bindAccountEmail( emailaddress ):
```

baseappemail

emailaddress string

```
def newPassword( oldpassword, newpassword ):
```

oldpassword string

newpassword string

```
def findEntity( entityID ):
```

ID

entityID int32ID

Entity

```
def getSpaceData( key ):
```

keyspace
space

setSpaceData

key string

stringkey



component

'cell', 'base', 'client', 'database', 'bot' 'editor'

entities

entities

Entities

entity_uuid

uuidIDID

entity_id

ID

spaceID

ID()

Entity

[KBEEngine]

Entity**KBEEngine** ...

```
import KBEEngine
```

```
def baseCall( self, methodName, methodArgs ):
```

```
def cellCall( self, methodName, methodArgs ):
```

```
def onDestroy( self ):
def onEnterWorld( self ):
def onLeaveWorld( self ):
def onEnterSpace( self ):
def onLeaveSpace( self ):
```

direction	Tuple of 3 floats as (roll, pitch, yaw)
id	Integer
position	Vector3
spaceID	uint32
isOnGround	bool
inWorld	bool
className	string



Entityclient

EntityMAILBOXbasecell .def

```
def baseCall( self, methodName, methodArgs ):
```

base
base

```
js: entity.baseCall("reqCreateAvatar", roleType, name);  
c#: entity.baseCall("reqCreateAvatar", new object[]{roleType, name});
```

methodName string
methodArgs objects

```
def cellCall( self, methodName, methodArgs ):
```

cell
cell

```
js: entity.cellCall("xxx", roleType, name);  
c#: entity.cellCall("xxx", new object[]{roleType, name});
```

methodName string
methodArgs objects



def onDestroy(self):

def onEnterWorld(self):

View
cellspace

def onLeaveWorld(self):

View
cellspace

def onEnterSpace(self):

space

def onLeaveSpace(self):

space

className

string

position

(x, y, z)

Vector3

direction

Entity

Vector3, (roll, pitch, yaw)

isOnGround

True **Entity**False

bool

KBEngine


KBEnginePython




Entity

```
def addSpaceGeometryMapping( spaceID, mapper, path,
shouldLoadOnServer, params ):
def addWatcher( path, dataType, getFunction ):
def address( ):
def MemoryStream( ):
def createEntity( entityType, spaceID, position, direction, params ):
def debugTracing( ):
def delSpaceData( spaceID, key ):
def delWatcher( path ):
def deregisterReadFileDescriptor( fileDescriptor ):
def deregisterWriteFileDescriptor( fileDescriptor ):
def executeRawDatabaseCommand( command, callback, threadID,
dbInterfaceName ):
def genUUID64( ):
def getResFullPath( res ):
def getSpaceData( spaceID, key ):
def getSpaceGeometryMapping( spaceID ):
def getWatcher( path ):
def getWatcherDir( path ):
def getAppFlags( ):
def hasRes( res ):
def isShuttingDown( ):
def listPathRes( path, extension ):
def matchPath( res ):
def open( res, mode ):
def publish( ):
def registerReadFileDescriptor( fileDescriptor, callback ):
def registerWriteFileDescriptor( fileDescriptor, callback ):
def raycast( spaceID, layer, src, dst ):
def reloadScript( fullReload ):
def scriptLogType( logType ):
```

```
def setAppFlags( flags ):  
def setSpaceData( spaceID, key, value ):  
def time( ):
```



```
def onCellAppData( key, value ):
def onCellAppDataDel( key ):
def onGlobalData( key, value ):
def onGlobalDataDel( key ):
def onInit( isReload ):
def onSpaceData( spaceID, key, value ):
def onSpaceGeometryLoaded( spaceID, mapping ):
def onAllSpaceGeometryLoaded( spaceID, isBootstrap, mapping ):
```



LOG_TYPE_DBG
LOG_TYPE_ERR
LOG_TYPE_INFO
LOG_TYPE_NORMAL
LOG_TYPE_WAR
NEXT_ONLY

cellAppData

component

entities

globalData

string

Entities

GlobalDataClient

```
def addSpaceGeometryMapping( spaceID, mapper, path,  
shouldLoadOnServer, params ):
```

```
    def onAllSpaceGeometryLoaded( self, spaceID, mappingName ):
```

```
3Drecastnavigation2DMapEditor
```

```
onAllSpaceGeometryLoaded() CellAppCellAppMgr
```

```
spaceID          uint32ID  
mapper          None  
path  
shouldLoadOnServer booleanTrue  
                  PyDictlayernavmesh  
params          KBEngine.addSpaceGeometryMapping(self.spaceID,  
None, resPath, True, {0 :  
"srv_xinshoucun_1.navmesh", 1 :  
"srv_xinshoucun.navmesh"})
```

```
def addWatcher( path, dataType, getFunction ):
```

```
>>> def countPlayers( ):
```

```
>>>     i = 0
>>>     for e in KBEEngine.entities.values():
>>>         if e.__class__.__name__ == "Avatar":
>>>             i += 1
>>>     return i
>>>
>>> KBEEngine.addWatcher( "players", "UINT32", countPlayers )
```

"scripts/players"countPlayers

path
dataType :
getFunction

```
def address( ):
```

```
def MemoryStream( ):
```

MemoryStream

MemoryStreamPythonKBEEngine

KBEEngine

```
>>> s = KBEEngine.MemoryStream()
>>> s
>>> b''
>>> s.append("UINT32", 1)
>>> s.pop("UINT32")
```

```
>>> 1
```

MemoryStream:

```
def createEntity( entityType, spaceID, position, direction, params ):
```

```
createEntityspace  
.def
```

```
# "thing"  
direction = ( 0, 0, thing.yaw )  
properties = { "open":1 }  
KBEngine.createEntity( "Door", thing.space, thing.position, direction,  
                        properties )
```

```
entityType string /scripts/entities.xml  
spaceID int32ID  
position 3float  
direction 3float(roll, pitch, yaw)  
params , Python Entity Entity
```

```
def debugTracing( ):
```

```
KBEnginePython  
EntityMailbox...
```

ERROR cellapp [0x0000cd64] [2014-11-12 00:38:07,300] -
PyGC::debugTracing(): FixedArray : leaked(128)
ERROR cellapp [0x0000cd64] [2014-11-12 00:38:07,300] -
PyGC::debugTracing(): EntityMailbox : leaked(8)

```
def delSpaceData( spaceID, key ):
```

```
    keyspace = space  
    setSpaceData
```

```
        spaceID int32ID  
        key string
```

```
def delWatcher( path ):
```

```
    path
```

```
def deregisterReadFileDescriptor( fileDescriptor ):
```

```
    KBEngine.registerReadFileDescriptor
```

```
    :
```

```
    http://www.kbengine.org/assets/other/py/Poller.py
```

```
    fileDescriptor socket/
```

```
def deregisterWriteFileDescriptor( fileDescriptor ):
```

KBEngine.registerWriteFileDescriptor

:

<http://www.kbengine.org/assets/other/py/Poller.py>

fileDescriptor socket/

```
def executeRawDatabaseCommand( command, callback, threadID,  
dbInterfaceName ):
```

command

MySQLSQL

4

```
def sqlcallback(result, rows, insertid, error):  
    print(result, rows, insertid, error)
```

```
result""  
DELETE
```

None

callback

```
rows""DELETE  
None
```

```
insertid""databaseID  
mysqlmysql_insert_id()mysql
```

```
error""None
```

threadID int32dbmgrthreadID

dbInterfaceName string, "default"kbengine_defaults.xml->dbmgr->databaseInterfaces

def genUUID64():

64ID
Cellappsgus
gus65535

ID
ID

64integer

def getResFullPath(res):

KBE_RES_PATH

res string

string

def getSpaceData(spaceID, key):

keyspace
space

setSpaceData

spaceID int32ID

key string

stringkey

```
def getSpaceGeometryMapping( spaceID ):
```

spaceID id

string

```
def getWatcher( path ):
```

KBEngine

baseapp1Python:

```
>>>KBEngine.getWatcher("/root/stats/runningTime")  
12673648533
```

```
>>>KBEngine.getWatcher("/root/scripts/players")  
32133
```

path string(GUIConsolewatcher)

```
def getWatcherDir( path ):
```

```
KBEngine()
```

```
baseapp1Python:
```

```
>>>KBEngine.getWatcher("/root")  
(stats, objectPools, network, syspaths, ThreadPool, cprofiles, scripts,  
numProxices, componentID, componentType, uid, numClients,  
globalOrder, username, load, gametime, entitiesSize, groupOrder)
```

```
    path string(GUIConsolewatcher)
```

```
    )
```

```
def getAppFlags( ):
```

```
APP, :                            KBEngine.setAppFlags
```

```
    KBEngine.APP_FLAGS_*
```

```
def hasRes( res ):
```

```
                            KBE_RES_PATH
```

```
:
```

```
>>>KBEngine.hasRes("scripts/entities.xml")  
True
```


res string

BOOL, True
False

```
def isShuttingDown( ):
```

```
onBaseAppShuttingDownTrue
```

```
TrueFalse
```

```
def listPathRes( path, extension ):
```

KBE_RES_PATH

:

```
>>>KBEngine.listPathRes("scripts/cell/interfaces")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", "txt")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", "txt|py")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", ("txt", "py"))  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

res string
extension string

Tuple,

```
def matchPath( res ):
```

[KBE_RES_PATH](#)

:

```
>>>KBEngine.matchPath("scripts/entities.xml")  
'/home/kbe/kbengine/demo/res/scripts/entities.xml'
```

res string()

string,

```
def open( res, mode ):
```

[KBE_RES_PATH](#)

res string
string
w
a (EOF ,

```
)  
r+  
mode w+ ( w )  
a+ ( a )  
rb  
wb ( w )  
ab ( a )  
rb+ ( r+ )  
wb+ ( w+ )  
ab+ ( a+ )
```

```
def publish( ):
```

```
int80debug1release
```

```
def raycast( spaceID, layer, src, dst ):
```

```
spacelayer
```

```
space addSpaceGeometryMapping
```

```
:
```

```
>>> KBEEngine.raycast( spaceID, entity.layer, (0, 10, 0), (0,  
(0.0000, 0.0000, 0.0000), (0.0000, 0.0000, 0.0000),  
(4.0000, 0.0000, 0.0000), (4.0000, 0.0000, 4.0000)), 0)
```

```
spaceID int32, spaceid
```

```
layer int8spacenavmeshnavmeshlayerlayer
```

list

```
def registerReadFileDescriptor( fileDescriptor, callback ):
```

```
:
```

```
http://www.kbengine.org/assets/other/py/Poller.py
```

```
    fileDescriptor socket/
```

```
    callback      socket/
```

```
def registerWriteFileDescriptor( fileDescriptor, callback ):
```

```
socket/
```

```
:
```

```
http://www.kbengine.org/assets/other/py/Poller.py
```

```
    fileDescriptor socket/
```

```
    callback      socket/
```

```
def reloadScript( fullReload ):
```

Python

1 **Cellapp**

2:

```
for e in KBEngine.entities.values():
    if type( e ) is Avatar.Avatar:
        e.customData.__class__ = CustomClass
```

KBEngine.onInit(True)

fullReload booleanFalseTrue

TrueFalse

```
def scriptLogType( logType ):
```

```
Python.print(: KBEngine.LOG_TYPE_*)
```

```
def setAppFlags( flags ):
```

APP

```
KBEngine.APP_FLAGS_NONE // ()
```

```
KBEngine.APP_FLAGS_NOT_PARTICIPATING_LOAD_BALANCING //
```

```
KBEngine.setAppFlags(KBEngine.APP_FLAGS_NOT_PARTICIPATING_LOAI  
| KBEngine.APP_FLAGS_*)
```

```
def setSpaceData( spaceID, key, value ):
```

keyspace

space **getSpaceData**

spaceID int32ID

key string
value string

def time():

uint32
>gameUpdateHertz

[kbengine.x](#)

```
def onCellAppData( key, value ):
```

```
KBEngine.cellAppData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

value

```
def onCellAppDataDel( key ):
```

```
KBEngine.cellAppData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

```
def onGlobalData( key, value ):
```

```
KBEngine.globalData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

value

```
def onGlobalDataDel( key ):
```

KBEngine.globalData
([kbengine_defaults.xml](#)->entryScriptFile)

key

def onInit(*isReload*):

([kbengine_defaults.xml](#)->entryScriptFile)

isReload bool

def onSpaceData(*spaceID*, *key*, *value*):

space
space

[setSpaceData](#)

spaceID ID

key

value

def onSpaceGeometryLoaded(*spaceID*, *mapping*):

[addSpaceGeometryMapping](#)

spaceID ID

mapping


```
def onAllSpaceGeometryLoaded( spaceID, isBootstrap, mapping ):
```

addSpaceGeometryMapping

spaceID ID

isBootstrap cellisBootstrapcell

mapping

LOG_TYPE_DBG

scriptLogType

LOG_TYPE_ERR

scriptLogType

LOG_TYPE_INFO

scriptLogType

LOG_TYPE_NORMAL

scriptLogType

LOG_TYPE_WAR

scriptLogType

NEXT_ONLY

Cellapp

cellAppData

CellApps CellApps

```
KBEngine.cellAppData[ "hello" ] = "there"
```

CellApp

```
print KBEngine.cellAppData[ "hello" ]
```

KBEngine.onCellApp

```
KBEngine.cellAppData[ "list" ] = [1, 2, 3]  
KBEngine.cellAppData[ "list" ][1] = 7
```

```
[1, 7, 3][1, 2, 3]
```

component

Python'cell', 'base', 'client', 'database', 'bot' 'editor'

entities

entities

[ghost](#)

destroy

```
>>> KBEngine.entities.garbage.items()  
[(1025, Avatar object at 0x7f92431ceae8.)]
```

```
>>> e = _[0][1]  
>>> import gc  
>>> gc.get_referents(e)  
[{'spacesIsOk': True, 'bootstrapIdx': 1}, ]
```

KBEnginePython

[KBEngine.debugTracing](#)

Entities

globalData

BaseAppsCellApps BaseAppsCellAppsCellAppMgr

```
KBEngine.globalData[ "hello" ] = "there"
```

CellappBaseapp

```
print KBEngine.globalData[ "hello" ]
```

[KBEngine.onGlobal](#)

```
KBEngine.globalData[ "list" ] = [1, 2, 3]
KBEngine.globalData[ "list" ][1] = 7
```

```
[1, 7, 3][1, 2, 3]
```

Entity

[KBEEngine]

Entity**KBEEngine** ...

```
import KBEEngine
```

```
def accelerate( self, accelerateType, acceleration ):
def addYawRotator( self, targetYaw, velocity, userArg ):
def addProximity( self, range, userArg ):
def addTimer( self, start, interval=0.0, userData=0 ):
def cancelController( self, controllerID ):
def clientEntity( self, destID ):
def canNavigate( self ):
def debugView( self ):
def delTimer( self, id ):
def destroy( self ):
def destroySpace( self ):
def entitiesInView( self ):
def entitiesInRange( self, range, entityType=None, position=None ):
def isReal( self ):
def moveToEntity( self, destEntityID, velocity, distance, userData,
faceMovement, moveVertically ):
def moveToPoint( self, destination, velocity, distance, userData,
faceMovement, moveVertically ):
def getViewRadius( self ):
def getViewHystArea( self ):
def getRandomPoints( self, centerPos, maxRadius, maxPoints, layer ):
def navigate( self, destination, velocity, distance, maxMoveDistance,
maxSearchDistance, faceMovement, layer, userData ):
def navigatePathPoints( self, destination, maxSearchDistance, layer ):
def setViewRadius( self, radius, hyst=5 ):
def teleport( self, nearbyMBRef, position, direction ):
def writeToDB( self, shouldAutoLoad, dbInterfaceName ):
```



```
def onDestroy( self ):
def onEnterTrap( self, entity, rangeXZ, rangeY, controllerID, userArg ):
def onEnteredView( self, entity ):
def onGetWitness( self ):
def onLeaveTrap( self, entity, rangeXZ, rangeY, controllerID, userArg ):
def onLoseControlledBy( self, id ):
def onLoseWitness( self ):
def onMove( self, controllerID, userData ):
def onMoveOver( self, controllerID, userData ):
def onMoveFailure( self, controllerID, userData ):
def onRestore( self ):
def onSpaceGone( self ):
def onTurn( self, controllerID, userData ):
def onTeleport( self ):
def onTeleportFailure( self ):
def onTeleportSuccess( self, nearbyEntity ):
def onTimer( self, timerHandle, userData ):
def onUpdateBegin( self ):
def onUpdateEnd( self ):
def onWitnessed( self, isWitnessed ):
def onWriteToDB( self ):
```

allClients	PyClient
base	BaseEntityMailBox
client	ClientEntityMailbox
controlledBy	BaseEntityMailBox
className	string
direction	Tuple of 3 floats as (roll, pitch, yaw)
hasWitness	boolean
id	Integer
isDestroyed	bool
isWitnessed	bool
layer	int8
otherClients	PyClient
position	Vector3
spaceID	uint32
topSpeed	float
topSpeedY	float
volatileInfo	float

Entitycell **Entity**"real""ghosted""ghost" **Entity**cell "real"
Entity"real" **Entity**0 "ghost" **Entity**

Entity teleport pythoncell
"trackEntity""turnToYaw" **Entity**"cancelController"

"View" **KBEngine** ViewView xz
EntityView **Entity** "setViewRadius"View "entitiesInRange"
"addProximity"

cellApp **EntityKBEngine.createEntity** baseApp
KBEngine.createCellEntity

EntityMAILBOXbaseclient .def

```
def accelerate( self, accelerateType, acceleration ):
```

```
Entity.moveToEntity  
Entity.moveToPoint  
Entity.navigate  
Entity.addYawRotator
```

```
accelerateType stringMovementTurn  
velocity float
```

```
def addYawRotator( self, targetYaw, velocity, userArg ):
```

```
yaw
```

```
Entity.onTurn
```

```
Entity.cancelControllerIDEntity.cancelController("Movement")
```

```
Entity.cancelController
```

```
targetYaw floatyaw  
velocity float  
userArg 0
```

0

```
def addProximity( self, rangeXZ, rangeY, userArg ):
```

xz

[Entity](#)
[Entity](#)onE

```
def onEnterTrap( self, entityEntering, rangeXZ, rangeY, cont  
def onLeaveTrap( self, entityLeaving, rangeXZ, rangeY, contr
```

[Entity.cancelControllerID](#)

addProximity()

[Entity.cancelController](#)

rangeXZ floatxz0
floaty0

[kbengine_defaults.xml](#)->cellapp

rangeY >rangemgr_y
yy
3D

userArg 0 0

id

```
def addTimer( self, start, interval=0.0, userData=0 ):
```

integer *onTimer*"initialOffset"1"repeatOffset"1"userArg"

onTimer entitycell21integertimeridtimer" *delTimer*"2
"userArg"

:

```
# addTimer
import KBEngine
```

```
class MyCellEntity( KBEngine.Entity ):
```

```
    def __init__( self ):
        KBEngine.Entity.__init__( self )
```

```
        # 51119
        self.addTimer( 5, 1, 9 )
```

```
        # 10
        self.addTimer( 1 )
```

```
    # Entity"onTimer"
```

```
    def onTimer( self, id, userArg ):
        print "MyCellEntity.onTimer called: id %i, userArg: %i" % (
            # if :
            #     self.delTimer( id )
```

initialOffset float

repeatOffset float

userArg integer"

onTimer"userArg

delT

:

integertimeridid *delTimer*

```
def cancelController( self, controllerID ):
```

cancelController

Entity real

controllerID controllerID /entity.cancelController(
"Movement")

def clientEntity(self, destID):

View

destID ID

def canNavigate(self):

Space [Entity.navigate](#) [real](#)
[Entity.addSpaceGeometryMapping](#)Navmesh2Dtile

bool SpaceTrue
False

def debugView(self):

debugView [Entity](#)Viewcell View [Entity](#)

```
INFO cellapp [0x00001a1c] [2014-11-04 00:28:41,409] - Avatar::debu  
INFO cellapp [0x00001a1c] [2014-11-04 00:28:41,409] - Avatar::debu  
INFO cellapp [0x00001a1c] [2014-11-04 00:28:41,409] - Avatar::debu  
INFO cellapp [0x00001a1c] [2014-11-04 00:28:41,409] - Avatar::debu  
INFO cellapp [0x00001a1c] [2014-11-04 00:28:41,409] - Avatar::debu
```

- #1000
- 4View

- 0View
- View 50.000
- View5.000

```
def delTimer( self, id ):
```

```
delTimer1delTimer delTimerid
```

```
    id integerid
```

```
def destroy( self ):
```

```
        Entity    ghost    ghostonDestroy()
```

```
def destroySpace( self ):
```

```
def entitiesInView( self ):
```

```
        View
```

```
def entitiesInRange( self, range, entityType=None, position=None ):
```

```
        3        View    cell
```

```
self.entitiesInRange( 100, 'Creature', (100, 0, 100) )
```


'Creature' Creature(100, 0, 100)100

```
[ e for e in self.entitiesInRange( 100, None, (100,0,100) ) if isi
```

'BaseType' BaseType'

range float

entityType

position **Vector3**,

Entity

```
def isReal( self ):
```

Entityrealghost

bool realTrue
False

```
def moveToEntity( self, destEntityID, velocity, distance, userData,  
faceMovement, moveVertically ):
```

Entity

ID

Entity.cancelController(movementID) **Entity.cancelController**(
"Movement")

```
def onMove( self, controllerID, userData ):  
def onMoveOver( self, controllerID, userData ):  
def onMoveFailure( self, controllerID, userData ):
```

:

Entity.cancelController

destEntityID int **EntityID**
velocity float **Entity**m/s
distance float0
userData objectuserData
faceMovement booltruefalse
moveVertically boolTrueFalse

intID

```
def moveToPoint( self, destination, velocity, distance, userData,  
faceMovement, moveVertically ):
```

Entity

ID

```
Entity.cancelController( movementID ) Entity.cancelController(  
"Movement" )
```

```
def onMove( self, controllerID, userData ):  
def onMoveOver( self, controllerID, userData ):  
def onMoveFailure( self, controllerID, userData ):
```

Entity.cancelController

destination Vector3 **Entity**
velocity float **Entity**m/s
distance float0
userData object
faceMovement booltruefalse
moveVertically booltruefalse

intID

```
def getViewRadius( self ):
```

EntityView

Entity.setViewRadius(radius, hyst)

float View

```
def getViewHystArea( self ):
```

EntityView

Entity.setViewRadius(radius, hyst)

float View

```
def getRandomPoints( self, centerPos, maxRadius, maxPoints, layer ):
```

Entity.navigate

centerPos Vector3 **Entity**
maxRadius float
maxPoints uint32
layer int8navmesh

tuple

```
def navigate( self, destination, velocity, distance, maxMoveDistance,  
maxSearchDistance, faceMovement, layer, userData ):
```

Entity

KBEngine

ID

Entity.cancelController(movementID) **Entity.cancelController**(
"Movement")

```
def onMove( self, controllerID, userData ):  
def onMoveOver( self, controllerID, userData ):  
def onMoveFailure( self, controllerID, userData ):
```

Entity.cancelController

destination Vector3 **Entity**
velocity float **Entity**m/s
distance float0

maxMoveDistance float
maxSearchDistance float
faceMovement booltruefalse
layer int8navmesh
userData object

intID

```
def navigatePathPoints( self, destination, maxSearchDistance, layer ):
```

Entitydestination

destination Vector3 **Entity**
maxSearchDistance float
layer int8navmesh

```
def setViewRadius( self, radius, hyst=5 ):
```

Entity

Witness

kbengine.xml'cellapp/defaultViewRadius'View

Entity.getViewRadius()**Entity.getViewHystArea()**

radius floatradiusView
hyst floatViewViewCPU
ViewViewViewView

None

```
def teleport( self, nearbyMRef, position, direction ):
```

Entity

CellMailbox mailbox

real

nearbyMRef **EntitySpaceCellMailbox** mailbox **Space**
Nonecell

position **Entity**3float(x, y, z)

direction **Entity**3float(roll,pitch, yaw)

```
def writeToDB( self, shouldAutoLoad, dbInterfaceName ):
```

basebaseonWriteToDB

cellbase

realbase

createBaseAnywhereFromDBIDbaseapp
baseapponBaseAppReady

shouldAutoLoad (kbengine_defaults.xml->baseapp->entryScriptFile)

def onAutoLoadEntityCreate(entityType, dbid):
 KBEngine.createBaseFromDBID(entityType, dbid)
string, "default"kbengine_defaults.xml-

dbInterfaceName >dbmgr->databaseInterfaces

```
def onDestroy( self ):
```

```
    Base.destroy()
```

```
def onEnterTrap( self, entity, rangeXZ, rangeY, controllerID, userArg ):
```

```
    Entity.addProximity
```

```
    entity
```

```
    rangeXZ    floatxz0
```

```
               floaty0
```

```
               kbengine_defaults.xml->cell
```

```
    rangeY     >rangemgr_y
```

```
               yy
```

```
               3D
```

```
    controllerID id
```

```
    userArg     addProximity
```

```
def onEnteredView( self, entity ):
```

```
    View
```

```
    entity View
```

```
def onGetWitness( self ):
```

```
    Witness
```

```
    Entity.hasWitness
```

```
def onLeaveTrap( self, entity, rangeXZ, rangeY, controllerID, userArg ):
```

```
    Entity.addProximity
```


entity

rangeXZ floatxz0
floaty0

[kbengine_defaults.xml](#)->cell

rangeY >rangemgr_y
yy
3D

controllerID ID

userArg [addProximity](#)

```
def onLoseControlledBy( self, id ):
```

[Entity.controlledBy](#)

id controlledByID

```
def onLoseWitness( self ):
```

[Witness](#)

[Entity.hasWitness](#)

```
def onMove( self, controllerID, userData ):
```

[Entity.moveToPointEntity.moveToEntityEntity.navigate](#)

controllerID ID

userData

```
def onMoveOver( self, controllerID, userData ):
```

[Entity.moveToPointEntity.moveToEntityEntity.navigate](#)

controllerID ID

userData

```
def onMoveFailure( self, controllerID, userData ):
```

```
    Entity.moveToPointEntity.moveToEntityEntity.navigate
```

```
    controllerID ID
```

```
    userData
```

```
def onRestore( self ):
```

```
CellCell
```

```
def onSpaceGone( self ):
```

```
    Space
```

```
def onTurn( self, controllerID, userData ):
```

```
    Entity.addYawRotatoryaw
```

```
    controllerID Entity.addYawRotatorID
```

```
    userData
```

```
def onTeleport( self ):
```

```
Base.teleport(Real entity)
```

```
cellteleport
```

```
Entity.tel
```

```
def onTeleportFailure( self ):
```

```
    Entity.teleport
```

```
def onTeleportSuccess( self, nearbyEntity ):
```

```
    Entity.teleport
```

```
    nearbyEntity
```

```
    Entity.teleportreal
```

```
def onTimer( self, timerHandle, userData ):
```

[Entity.addTin](#)

timerHandle id

userData [Entity.addTimer](#) integer

```
def onUpdateBegin( self ):
```

```
def onUpdateEnd( self ):
```

```
def onWitnessed( self, isWitnessed ):
```

WitnessView

AIAI

isWitnessed bool True False

[Entity.isWitnessed](#)

```
def onWriteToDB( self ):
```

allClients

View()

avatarViewABC

avatar.allClients.attack(monsterIDskillID, damage)

ABattack

[Entity.clientEntity](#)

[Entity.otherClients](#)

base

base[Base](#)mailbox

[Base](#)None

[Entity.clientEntity](#)

[Entity.allClients](#)

[Entity.otherClients](#)

MAILBOX

className

string

client

clientmailboxNone

Entity.clientEntity
Entity.allClients
Entity.otherClients

MAILBOX

controlledBy

BaseEntityMailBoxNone giveClientToBaseEntityMailBox

Entity.onLoseControlledBy

BaseEntityMailBox

direction

Entity

self.direction.y = 1.0 self.direction.z = 1.0

Vector3, (roll, pitch, yaw)

hasWitness

True **Witness**WitnessViewFalse

bool

id

idEntityididbasecellclient

int32

isDestroyed

True [Entity](#)

bool

isOnGround

True [Entity](#)False

bool

isWitnessed

WitnessView, TrueFalse

[Entity.onWitnessed](#)

bool

layer

spacenavmeshnavmeshlayerlayer layer

[KBEngine.addSpaceGeometryMapping](#)

int8

otherClients

View()

avatarViewABC

avatar.otherClients.attack(monsterIDskillID, damage)

ABAttack

Entity.clientEntity

Entity.otherClients

position

(x, y, z)

```
self.position.y = 10.0
```

```
import Math  
self.copyPosition = Math.Vector3( self.position )
```

Vector3

spaceID

IDcell

Integer

topSpeed

xz/

Entity.topSpeedY

float

topSpeedY

y/

Entity.topSpeed

float

volatileInfo

Entity

positiondirection

floatpositionyawpitchrollView

booloptimizedY

true(navigate)Ytrue

.def

```
<Volatile>
  <position/>           <!-- -->
  <yaw/>                <!-- -->
  <pitch>20</pitch>    <!-- 20 -->
  <optimized> true </optimized>
</Volatile>           <!-- roll -->
```

sequence (float, float, float,
float)

KBEngine

KBEnginePython



Base
Proxy

```
def addWatcher( path, dataType, getFunction ):
def address( ):
def MemoryStream( ):
def charge( ordersID, dbID, byteDatas, pycallback ):
def createBase( ):
def createBaseAnywhere( entityType, *params, callback ):
def createBaseRemotely( entityType, baseMB, *params, callback ):
def createBaseFromDBID( entityType, dbID, callback, dbInterfaceName ):
def createBaseAnywhereFromDBID( entityType, dbID, callback,
dbInterfaceName ):
def createBaseRemotelyFromDBID( entityType, dbID, baseMB, callback,
dbInterfaceName ):
def createBaseLocally( entityType, *params ):
def createEntity( ):
def debugTracing( ):
def delWatcher( path ):
def deleteBaseByDBID( entityType, dbID, callback, dbInterfaceName ):
def deregisterReadFileDescriptor( fileDescriptor ):
def deregisterWriteFileDescriptor( fileDescriptor ):
def executeRawDatabaseCommand( command, callback, threadID,
dbInterfaceName ):
def genUUID64( ):
def getResFullPath( res ):
def getWatcher( path ):
def getWatcherDir( path ):
def getAppFlags( ):
def hasRes( res ):
def isShuttingDown( ):
def listPathRes( path, extension ):
def lookUpBaseByDBID( entityType, dbID, callback, dbInterfaceName ):
def matchPath( res ):
```

```
def open( res, mode ):
def publish( ):
def quantumPassedPercent( ):
def registerReadFileDescriptor( fileDescriptor, callback ):
def registerWriteFileDescriptor( fileDescriptor, callback ):
def reloadScript( fullReload ):
def scriptLogType( logType ):
def setAppFlags( flags ):
def time( ):
```



```
def onBaseAppReady( isBootstrap ):
def onBaseAppShutDown( state ):
def onCellAppDeath( addr ):
def onFini( ):
def onBaseAppData( key, value ):
def onBaseAppDataDel( key ):
def onGlobalData( key, value ):
def onGlobalDataDel( key ):
def onInit( isReload ):
def onLoseChargeCB( orderID, dbID, success, datas ):
def onReadyForLogin( isBootstrap ):
def onReadyForShutDown( ):
def onAutoLoadEntityCreate( entityType, dbID ):
```

LOG_ON_ACCEPT
LOG_ON_REJECT
LOG_ON_WAIT_FOR_DESTROY
LOG_TYPE_DBG
LOG_TYPE_ERR
LOG_TYPE_INFO
LOG_TYPE_NORMAL
LOG_TYPE_WAR
NEXT_ONLY

component

string

entities

Entities

baseAppData

GlobalDataClient

globalData

GlobalDataClient

```
def addWatcher( path, dataType, getFunction ):
```

```
>>> def countPlayers( ):
>>>     i = 0
>>>     for e in KBEEngine.entities.values():
>>>         if e.__class__.__name__ == "Avatar":
>>>             i += 1
>>>     return i
>>>
>>> KBEEngine.addWatcher( "players", "UINT32", countPlayers )
```

"scripts/players"countPlayers

path
dataType :
getFunction

```
def address( ):
```

```
def MemoryStream( ):
```

MemoryStream

MemoryStreamPythonKBEngine

KBEngine

```
>>> s = KBEngine.MemoryStream()  
>>> s  
>>> b''  
>>> s.append("UINT32", 1)  
>>> s.pop("UINT32")  
>>> 1
```

MemoryStream:

```
def charge( ordersID, dbID, byteDatas, pycallback ):
```

ordersID stringID

dbID uint64 **databaseID**

byteDatas bytes

: (interfacesKBEngine.chargeResponse)

def on**ChargeCB(self, orderID, dbID, success, datas):

pycallback ordersIDstringID

dbIDuint64entity **databaseID**

successbool

datasbytes

```
def createBase( ):
```

KBEngine.createBaseLocally.

```
def createBaseAnywhere( entityType, params, callback ):
```

Base BaseappBase

KBEngine.createBaseLocally Baseapp

Python

Python".def"

```
params = {  
    "name" : "kbe", # base, BASE_AND_CLIENT  
    "HP" : 100,    # cell, ALL_CLIENT, in cellData  
    "tmp" : "tmp"    # baseEntity.tmp  
}  
  
def onCreateBaseCallback(entity)  
    print(entity)  
  
createBaseAnywhere("Avatar", params, onCreateBaseCallback)
```

entityType string **Base** /scripts/entities.xml
params , Python **Base** **Base**
 cellData Python cell
callback callback

Basemailbox

```
def createBaseRemotely( entityType, baseMB, params, callback ):
```

baseMBbaseapp

Base

callback baseRef **mailboxBase**databaseIDID
wasActivewasActiveTruebaseRef()baseRef
None databaseID0wasActiveFalse
ID

dbInterfaceName string, "default"kbengine_defaults.xml-
>dbmgr->databaseInterfaces

```
def createBaseAnywhereFromDBID( entityType, dbID, callback,
dbInterfaceName ):
```

Base BaseappBase

BaseApps

Base

entityType stringBase /**scripts/**
dbID IDID
3baseRefdatabaseIDwasActive
baseRef **mailboxBase**databaseIDID
callback wasActivewasActiveTruebaseRef()baseRef
None databaseID0wasActiveFalse
ID

dbInterfaceName string, "default"kbengine_defaults.xml-
>dbmgr->databaseInterfaces

Basemailbox

```
def createBaseRemotelyFromDBID( entityType, dbID, baseMB, callback,
dbInterfaceName ):
```

baseMBbaseapp

Base

Base

entityType	stringBase	/scripts/
dbID	IDID	
baseMB	BaseMailbox	BaseMailboxBaseBaseapp
	3baseRefdatabaseIDwasActive	
	baseRef	mailboxBase databaseIDID
callback	wasActivewasActiveTruebaseRef()baseRef	
	NonedatabaseID0wasActiveFalse	
	ID	
dbInterfaceName	string, "default"kbengine_defaults.xml->dbmgr->databaseInterfaces	

Basemailbox

```
def createBaseLocally( entityType, params ):
```

Base

Python

Python".def"

KBEngine.createBaseAnywhere **Baseapp**

```
params = {
    "name" : "kbe", # base, BASE_AND_CLIENT
    "HP" : 100,    # cell, ALL_CLIENT, in cellData
    "tmp" : "tmp"  # baseEntity.tmp
}

baseEntity = createBaseLocally("Avatar", params)
```

entityType stringBase
params , Python
'*cellData*'Python cell

/scripts/entities
Base Base

Base **Base**

```
def createEntity( ):
```

KBEngine.createBaseLocally.

```
def debugTracing( ):
```

KBEnginePython
EntityMailbox...

ERROR cellapp [0x0000cd64] [2014-11-12 00:38:07,300] -
PyGC::debugTracing(): FixedArray : leaked(128)
ERROR cellapp [0x0000cd64] [2014-11-12 00:38:07,300] -
PyGC::debugTracing(): EntityMailbox : leaked(8)

path

```
def delWatcher( path ):
```

path

```
def deleteBaseByDBID( entityType, dbID, callback, dbInterfaceName ):
```


<i>command</i>	MySQLSQL	
	4	
	def sqlcallback(result, rows, insertid, error): print(result, rows, insertid, error)	
<i>callback</i>	result"" DELETE	None
	rows""DELETE None	
	insertid""databaseID mysqlmysql_insert_id()mysql	
	error""None	
<i>threadID</i>	int32dbmgrthreadIDID	
<i>dbInterfaceName</i>	string, "default"kbengine_defaults.xml->dbmgr->databaseInterfaces	

```
def genUUID64( ):
```

```
64ID
Baseappsgus
gus65535
```

ID
ID

64integer

```
def getResFullPath( res ):
```

KBE_RES_PATH

res string

string

```
def getWatcher( path ):
```

KBEngine

baseapp1Python:

```
>>>KBEngine.getWatcher("/root/stats/runningTime")  
12673648533
```

```
>>>KBEngine.getWatcher("/root/scripts/players")  
32133
```

path string(GUIConsolewatcher)

```
def getWatcherDir( path ):
```

KBEngine()

baseapp1Python:

```
>>>KBEngine.getWatcher("/root")
('stats', 'objectPools', 'network', 'syspaths', 'ThreadPool', 'cprofiles', 'scripts',
'numProxies', 'componentID', 'componentType', 'uid', 'numClients',
'globalOrder', 'username', 'load', 'gametime', 'entitiesSize', 'groupOrder')
```

path string(GUIConsolewatcher)

()

```
def getAppFlags( ):
```

APP, : [KBEngine.setAppFlags](#)

KBEngine.APP_FLAGS_*

```
def hasRes( res ):
```

[KBE_RES_PATH](#)

:

```
>>>KBEngine.hasRes("scripts/entities.xml")
True
```

res string

BOOL, True
False

```
def isShuttingDown():
```

```
onBaseAppShutDown(state=0)True
```

TrueFalse

```
def listPathRes( path, extension ):
```

KBE_RES_PATH

:

```
>>>KBEngine.listPathRes("scripts/cell/interfaces")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", "txt")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", "txt|py")  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

```
>>>KBEngine.listPathRes("scripts/cell/interfaces", ("txt", "py"))  
('/home/kbe/kbengine/demo/res/scripts/cell/interfaces/AI.py',  
'/home/kbe/kbengine/demo/res/scripts/cell/interfaces/.txt')
```

res string
extension string

Tuple,

```
def lookUpBaseByDBID( entityType, dbID, callback, dbInterfaceName ):
```

KBEngineBasemailbox

entityType stringBase /scripts/
dbID BaseIDID databaseID
callback callbackTrue
dbInterfaceName string, "default"kbengine_defaults.xml-
>dbmgr->databaseInterfaces

```
def matchPath( res ):
```

KBE_RES_PATH

:

```
>>>KBEngine.matchPath("scripts/entities.xml")  
'/home/kbe/kbengine/demo/res/scripts/entities.xml'
```

res string()

string,

```
def open( res, mode ):
```

KBE_RES_PATH

```
    res    string  
          string  
          w  
          a ( EOF ,  
            )  
          r+  
          w+ ( w )  
    mode  a+ ( a )  
          rb  
          wb ( w )  
          ab ( a )  
          rb+ ( r+ )  
          wb+ ( w+ )  
          ab+ ( a+ )
```

```
def publish( ):
```

```
int80debug1release
```

```
def quantumPassedPercent( ):
```

```
tick
```

tick

```
def registerReadFileDescriptor( fileDescriptor, callback ):
```

```
:
```

```
http://www.kbengine.org/assets/other/py/Poller.py
```

```
    fileDescriptor socket/
```

```
    callback      socket/
```

```
def registerWriteFileDescriptor( fileDescriptor, callback ):
```

```
socket/
```

```
:
```

```
http://www.kbengine.org/assets/other/py/Poller.py
```

```
    fileDescriptor socket/
```

```
    callback      socket/
```

```
def reloadScript( fullReload ):
```

Python

1 **Baseapp**

2:

```
for e in KBEngine.entities.values():
    if type( e ) is Avatar.Avatar:
        e.customData.__class__ = CustomClass
```

`KBEngine.onInit(True)`

`fullReload` booleanFalseTrue

TrueFalse

```
def scriptLogType( logType ):
```

```
Python.print(: KBEngine.LOG_TYPE_*)
```

```
def setAppFlags( flags ):
```

APP

```
KBEngine.APP_FLAGS_NONE // ()
```

```
KBEngine.APP_FLAGS_NOT_PARTICIPATING_LOAD_BALANCING //
```

```
KBEngine.setAppFlags(KBEngine.APP_FLAGS_NOT_PARTICIPATING_LOAI  
| KBEngine.APP_FLAGS_*)
```

```
def time( ):
```

uint32

>gameUpdateHertz

[kbengine.x](#)



```
def onBaseAppReady( isBootstrap ):
```

```
    Baseapp  
(    kbengine_defaults.xml->entryScriptFile)
```

```
    isBootstrap bool                Baseapp
```

```
def onBaseAppShutdown( state ):
```

```
Baseapp  
(    kbengine_defaults.xml->entryScriptFile)
```

```
    state state0state1state2
```

```
def onCellAppDeath( addr ):
```

```
cellapp  
(    kbengine_defaults.xml->entryScriptFile)
```

```
    cellapp  
    addr tuple:(ip, port)
```

```
def onFini( ):
```

[kbengine_defaults.xml](#)->entryScriptFile)

```
def onBaseAppData( key, value ):
```

```
KBEngine.baseAppData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

value

```
def onBaseAppDataDel( key ):
```

```
KBEngine.baseAppData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

```
def onGlobalData( key, value ):
```

```
KBEngine.globalData  
( kbengine\_defaults.xml->entryScriptFile)
```

key

value

```
def onGlobalDataDel( key ):
```

KBEngine.globalData
([kbengine_defaults.xml](#)->entryScriptFile)

key

def onInit(*isReload*):

([kbengine_defaults.xml](#)->entryScriptFile)

isReload bool

def onLoseChargeCB(*orderID*, *dbID*, *success*, *datas*):

interfacesKBEngine.chargeResponseinterfaces
([kbengine_defaults.xml](#)->entryScriptFile)

ordersID stringID

dbID uint64ID, :
[Base.databaseID](#)

success bool

datas bytes

def onReadyForLogin(*isBootstrap*):

loginapp
([kbengine_defaults.xml](#)->entryScriptFile)

isBootstrap bool

[Baseapp](#)

1.00.0~1.0

def onReadyForShutDown():

([kbengine_defaults.xml](#)->entryScriptFile)

boolTrue

def onAutoLoadEntityCreate(*entityType*, *dbID*):

createBaseAnywhereFromDBID

[Base.writeToDB](#)

onBaseAppReadyonBaseAppReady

entityType stringBase

[/scripts/entities](#)

dbID

[BaseIDID](#) [databaseID](#)

LOG_ON_ACCEPT

[Proxy.onLogOnAttempt](#)client [Proxy](#)
[Proxy](#)clientclient

LOG_ON_REJECT

[Proxy.onLogOnAttempt](#)client [Proxy](#)

LOG_ON_WAIT_FOR_DESTROY

[Proxy.onLogOnAttempt](#)client [Proxy](#) [Proxy.destroy](#)
[Proxy.destroy](#)CellEntity

LOG_TYPE_DBG

[scriptLogType](#)

LOG_TYPE_ERR

[scriptLogType](#)

LOG_TYPE_INFO

scriptLogType

LOG_TYPE_NORMAL

scriptLogType

LOG_TYPE_WAR

scriptLogType

NEXT_ONLY

Base.shouldAutoBackupBase.shouldAutoArchiveFalse0

component

Python'cell', 'base', 'client', 'database', 'bot' 'editor'

entities

entities

destroy

```
>>> KBEngine.entities.garbage.items()  
[(1025, Avatar object at 0x7f92431ceae8.)]
```

```
>>> e = _[0][1]
>>> import gc
>>> gc.get_referents(e)
[{'spacesIsOk': True, 'bootstrapIdx': 1}, ]
```

KBEnginePython
[KBEngine.debugTracing](#)

Entities

baseAppData

BaseApps BaseApps

```
KBEngine.baseAppData[ "hello" ] = "there"
```

BaseApps

```
print KBEngine.baseAppData[ "hello" ]
```

[KBEngine.onBaseAp](#)

```
KBEngine.baseAppData[ "list" ] = [1, 2, 3]
KBEngine.baseAppData[ "list" ][1] = 7
```

```
[1, 7, 3][1, 2, 3]
```


globalData

BaseAppsCellApps BaseAppsCellApps

```
KBEngine.globalData[ "hello" ] = "there"
```

BaseappCellapp

```
print KBEngine.globalData[ "hello" ]
```

KBEngine.onGlobalI

```
KBEngine.globalData[ "list" ] = [1, 2, 3]  
KBEngine.globalData[ "list" ][1] = 7
```

```
[1, 7, 3][1, 2, 3]
```



Base

[KBEEngine]

Base**KBEEngine** ...

```
import KBEEngine
```

```
def addTimer( self, initialOffset, repeatOffset=0, userArg=0 ):
def createCellEntity( self, cellEntityMB ):
def createInNewSpace( self, cellappIndex ):
def delTimer( self, id ):
def destroy( self, deleteFromDB, writeToDB ):
def destroyCellEntity( self ):
def teleport( self, baseEntityMB ):
def writeToDB( self, callback, shouldAutoLoad, dbInterfaceName ):
```



```
def onCreateCellFailure( self ):
def onDestroy( self ):
def onGetCell( self ):
def onLoseCell( self ):
def onPreArchive( self ):
def onRestore( self ):
def onTimer( self, timerHandle, userData ):
def onWriteToDB( self, cellData ):
```

cell	CellEntityMailbox
cellData	CELLDATADICT
className	string
client	ClientEntityMailbox
databaseID	int64
databaseInterfaceName	string
id	int32
isDestroyed	bool
shouldAutoArchive	True, False or KBEngine.NEXT_ONLY
shouldAutoBackup	True, False or KBEngine.NEXT_ONLY

BaseBaseapp **Base KBEngine.createBase(createBase)** **Base**
CellappKBEngine.createEntityOnBaseApp

Basecells **cell** **cellbase**
CellEntityMailBoxbasecell **cellcell** **cellKBEngine**

```
def addTimer( self, initialOffset, repeatOffset=0, userArg=0 ):
```

```
    onTimer"initialOffset"1"repeatOffset"1"userArg"  
integer
```

```
    onTimer entitybase21integertimeridtimer"    delTimer"2  
    "userArg"
```

```
:
```

```
# addTimer  
import KBEngine
```

```
class MyBaseEntity( KBEngine.Base ):
```

```
    def __init__( self ):  
        KBEngine.Base.__init__( self )
```

```
        # 51119  
        self.addTimer( 5, 1, 9 )
```

```
        # 10  
        self.addTimer( 1 )
```

```
    # Base"onTimer"
```

```
    def onTimer( self, id, userArg ):  
        print "MyBaseEntity.onTimer called: id %i, userArg: %i" % (   
            # if :  
            #     self.delTimer( id )
```

```
    initialOffset float
```

```
    repeatOffset float
```

```
    userArg integer"
```

```
    onTimer"userArg
```

```
    delT
```

```
:
```

```
integer"timeridid    delTimer
```



```
def createCellEntity( self, cellEntityMB ):
```

```
    cell
```

```
    cellcellData.def  
(roll, pitch, yaw)"position", "direction" "spaceID"
```

```
        CellEntityMailBox cell
```

```
            CellEntityMailBoxBaseMailboxbaseMailbox  
                base CellEnti
```

```
                baseMailboxOfNearbyEntity.createCellNearSelf( self )  
cellEntityMB  
                base  
                def createCellNearSelf( self, baseMailbox ):  
                    baseMailbox.createCellNearHere( self.cell )  
                basecreateCellNearSelf()  
                def createCellNearHere( self, cellMailbox ):  
                    self.createCellEntity( cellMailbox )
```

```
def createInNewSpace( self, cellappIndex ):
```

```
    cellcellappmgr
```

```
    cellcellData.def  
(roll, pitch, yaw)"position", "direction" "spaceID"
```

```
        integerNone00cellappspace  
        4cellappcellappIndex1234
```

cellappIndex cellapp43cellappIndex44152

KBEngine.setAppFlags(KBEngine.APP_FLAGS_NOT_PARTCII
spacecellappcellappcellappspace spacecellappIndex0None

def delTimer(self, id):

delTimer1delTimer delTimerid

Base.addTimer

id integerid

def destroy(self, deleteFromDB, writeToDB):

basecellcellcell

onLoseCellself.destroybase

deleteFromDB TrueFalse

writeToDB True

def destroyCellEntity(self):

*destroyCellEntity*cellcell

def teleport(self, baseEntityMB):

teleportcell

Entity.onTeleportSuccess

baseEntityMB baseEntityMB

mai

```
def writeToDB( self, callback, shouldAutoLoad, dbInterfaceName ):
```

callback booleanbase

```
createBaseAnywhereFromDBIDbaseapp  
baseapponBaseAppReady
```

shouldAutoLoad (kbengine_defaults.xml->baseapp->entryScriptFile)

```
def onAutoLoadEntityCreate(entityType, dbid):  
    KBEngine.createBaseFromDBID(entityType, dbid)
```

dbInterfaceName string, "default"kbengine_defaults.xml->
>dbmgr->databaseInterfaces

def onCreateCellFailure(self):

cell

def onDestroy(self):

Base.destroy()

def onGetCell(self):

cell

def onLoseCell(self):

cell

def onPreArchive(self):

Bas

FalseTrue

def onRestore(self):

BaseBase

```
def onTimer( self, timerHandle, userData ):
```

[Base.addTime](#)

timerHandle id

userData [Base.addTimer](#) integer

```
def onWriteToDB( self, cellData ):
```

writeToDB

cellData cell

[cellData](#)

cell

cellcell **MAILBOX**basecellNone

MAILBOX

cellData

cellDatabasecellcell

cell **cellData**cellposition, direction and spaceID

CELLDATADICT

className

string

client

clientmailboxbaseNone

MAILBOX

databaseID

databaseIDID(id)iduint6400

int64

databaseInterfaceName

databaseInterfaceNamekbengine_defaults->dbmgrdatabaseID>0

string

id

idididbasecellclient

int32

isDestroyed

BaseTrue

bool

shouldAutoArchive

TrueFalse

True, False or [KBEngine.NEXT_ONLY](#)

shouldAutoBackup

TrueFalse

True, False or [KBEngine.NEXT_ONLY](#)

Proxy


[KBEngine]

Proxy**KBEngine** ...


```
import KBEngine
```



Base



```
def disconnect( self ):
def getClientType( self ):
def getClientDatas( self ):
def giveClientTo( self, proxy ):
def streamFileToClient( self, resourceName, desc="", id=-1 ):
def streamStringToClient( self, data, desc="", id=-1 ):
```



```
def onClientDeath( self ):
def onClientGetCell( self ):
def onEntitiesEnabled( self ):
def onGiveClientToFailure( self ):
def onLogOnAttempt( self, ip, port, password ):
def onStreamComplete( self, id, success ):
```

__ACCOUNT_NAME__ string

__ACCOUNT_PASSWORD__ string

clientAddr

entitiesEnabled bool

hasClient bool

roundTripTime

timeSinceHeardFromClient



ProxyBaseBase

Proxy

```
def disconnect( self ):
```

```
def getClientType( self ):
```

```
:  
    UNKNOWN_CLIENT_COMPONENT_TYPE = 0,  
    CLIENT_TYPE_MOBILE = 1, //  
    CLIENT_TYPE_WIN = 2, // pc exe  
    CLIENT_TYPE_LINUX = 3 // Linux Application program  
    CLIENT_TYPE_MAC = 4 // Mac Application program  
    CLIENT_TYPE_BROWSER = 5, // web html5flash  
    CLIENT_TYPE_BOTS = 6, // bots  
    CLIENT_TYPE_MINI = 7, // Mini-Client  
    CLIENT_TYPE_END = 8 // end
```

```
def getClientDatas( self ):
```

```
interfaces
```

```
:  
    tuple, 2tuple(bytesbytes)datas datasbytes
```

```
def giveClientTo( self, proxy ):
```


ProxyProxyProxy

Proxy.onGiveClientToFailure

proxy

```
def streamFileToClient( self, resourceName, desc="", id=-1 ):
```

streamStringToClient()

Proxy.onStreamComplete

resourceName

desc

id 16id-1idid

id

```
def streamStringToClient( self, data, desc="", id=-1 ):
```

16id
IDidid

ProxyonStreamComplete

Proxy.onStreamCompleteEntity.onStreamDataStarted
Entity.onStreamDataRecvEntity.onStreamDataCompleted

data

desc

id 16id-1id

id

```
def onClientDeath( self ):
```

```
def onClientGetCell( self ):
```

```
    cell
```

```
def onEntitiesEnabled( self ):
```

```
giveClientTo
```

```
def onGiveClientToFailure( self ):
```

```
    giveClientTo
```

```
def onLogOnAttempt( self, ip, port, password ):
```

```
AB
```

```
KBEngine.LOG_ON_ACCEPT
```

```
KBEngine.LOG_ON_REJECT
```

```
KBEngine.LOG_ON_WAIT_FOR_DESTROY
```

```
    ip      IP
```

```
    port
```

```
    password MD5
```

```
def onStreamComplete( self, id, success ):
```

```
    Proxy.streamStringToClient()Proxy.streamFileToClient()
```

id id
success

__ACCOUNT_NAME__

proxy__ACCOUNT_NAME__

__ACCOUNT_PASSWORD__

proxy__ACCOUNT_PASSWORD__MD5

clientAddr

tupleip

entitiesEnabled

hasClient

Proxy

roundTripTime

ProxyLinux

timeSinceHeardFromClient


KBEngine

KBEnginePythonloginapp



```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
def delTimer( id ):
```

```
def onLoginAppReady( ):
def onLoginAppShutDown( ):
def onRequestLogin( loginName, password, clientType, datas ):
def onLoginCallbackFromDB( loginName, accountName, errorno, datas ):
def onRequestCreateAccount( accountName, password, datas ):
def onCreateAccountCallbackFromDB( accountName, errorno, datas ):
```

```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
callbackObj"initialOffset"1"repeatOffset"1
```

```
:
```

```
# addTimer
```

```
import KBEngine
```

```
# 51119
```

```
KBEngine.addTimer( 5, 1, onTimer_Callbackfun )
```

```
# 10
```

```
KBEngine.addTimer( 1, onTimer_Callbackfun )
```

```
def onTimer_Callbackfun( id ):
```

```
print "onTimer_Callbackfun called: id %i" % ( id )
```

```
# if :
```

```
# KBEngine.delTimer( id )
```

initialOffset float

repeatOffset float

callbackObj function

delT

```
:
```

```
integertimeridid delTimer
```

```
def delTimer( id ):
```

```
delTimer1delTimer delTimerid
```

```
KBEngine.addTimer
```

id integerid

```
def onLoginAppReady( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onLoginAppShutDown( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onRequestLogin( loginName, password, clientType, datas ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
loginName string  
password stringMD5  
clientType integer  
datas bytes
```

```
TupledatasKBEngine.SERVER_SUCCESS,  
loginName, password, clientType, datas
```

```
def onLoginAppReady( ):
```

([kbengine_defaults.xml](#)->entryScriptFile)

```
def onLoginAppShutDown( ):
```

([kbengine_defaults.xml](#)->entryScriptFile)

```
def onLoginCallbackFromDB( loginName, accountName, errorno, datas ):
```

dbmgr

([kbengine_defaults.xml](#)->entryScriptFile)

loginName string
accountName stringdbmgr
errorno integerKBEEngine.SERVER_SUCCESS
datas bytesdbmgrinterfaces

```
def onRequestCreateAccount( accountName, password, data ):
```

([kbengine_defaults.xml](#)->entryScriptFile)

accountName string
password stringMD5

datas bytes

Tuple*datas*KBEngine.SERVER_SUCCESS,
loginName, password, *datas*

```
def onCreateAccountCallbackFromDB( accountName, errorno, datas ):
```

dbmgr

([kbengine_defaults.xml](#)->entryScriptFile)

accountName string

errorno integerKBEngine.SERVER_SUCCESS

datas bytesdbmgrinterfaces


KBEngine

KBEnginePythondbmgr



```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
def delTimer( id ):
```



```
def onDBMgrReady( ):
def onDBMgrShutDown( ):
def onReadyForShutDown( ):
def onSelectAccountDBInterface( accountName ):
```

```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
callbackObj"initialOffset"1"repeatOffset"1
```

```
:
```

```
# addTimer
```

```
import KBEngine
```

```
# 51119
```

```
KBEngine.addTimer( 5, 1, onTimer_Callbackfun )
```

```
# 10
```

```
KBEngine.addTimer( 1, onTimer_Callbackfun )
```

```
def onTimer_Callbackfun( id ):
```

```
print "onTimer_Callbackfun called: id %i" % ( id )
```

```
# if :
```

```
# KBEngine.delTimer( id )
```

initialOffset float

repeatOffset float

callbackObj function

delT

```
:
```

```
integertimeridid delTimer
```

```
def delTimer( id ):
```

```
delTimer1delTimer delTimerid
```

```
KBEngine.addTimer
```

id integerid

```
def onDBMgrReady( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onDBMgrShutDown( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onReadyForShutDown( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
    boolTrue
```

```
def onSelectAccountDBInterface( accountName ):
```

```
    dbmgr
```

```
        kbengine\_defaults.xml->dbmgr->databaseInterfaces  
    accountName
```

([kbengine_defaults.xml](#)->entryScriptFile)

accountName string

string

[kbengine_defaults.xml](#)->dbmgr->databaseIn

KBEngine


KBEngine

KBEngine,




Entity

PyClientApp



```
def addBots( reqCreateAndLoginTotalCount,  
reqCreateAndLoginTickCount=0, reqCreateAndLoginTickTime=0 ):  
def callback( initialOffset, callbackObj ):  
def cancelCallback( id ):  
def genUUID64( ):  
def getWatcher( path ):  
def getWatcherDir( path ):  
def scriptLogType( logType ):
```



```
def onInit( isReload ):
```

```
def onFinish( ):
```



bots **bots**

component
string

```
def addBots( reqCreateAndLoginTotalCount,  
reqCreateAndLoginTickCount=0, reqCreateAndLoginTickTime=0 ):
```

```
:
```

```
# addBots  
    import KBEngine  
  
    # 5  
    KBEngine.addBots( 5 )  
  
    # 10005(s)  
    KBEngine.addBots( 1000, 5, 10 )
```

```
reqCreateAndLoginTotalCount integer  
reqCreateAndLoginTickCount integer  
reqCreateAndLoginTickTime integer()
```

```
def callback( initialOffset, callbackObj ):
```

```
callbackObj"initialOffset"
```

```
:
```

```
# callback  
    import KBEngine  
  
    # 1  
    KBEngine.callback( 1, onCallbackfun )  
  
    def onCallbackfun( ):  
        print "onCallbackfun called"
```

initialOffset float
callbackObj function

:
integercallbackidid **cancelCallback**

def cancelCallback(*id*):

cancelCallback cancelCallbackid

KBEngine.callback

id integerid

def genUUID64():

64ID
Cellappsgus
gus65535

ID
ID

64integer

def getWatcher(*path*):

KBEngine

baseapp1Python:

```
>>>KBEngine.getWatcher("/root/stats/runningTime")  
12673648533
```

```
>>>KBEngine.getWatcher("/root/scripts/players")  
32133
```

path string(GUIConsolewatcher)

```
def getWatcherDir( path ):
```

KBEngine()

baseapp1Python:

```
>>>KBEngine.getWatcher("/root")  
( 'stats', 'objectPools', 'network', 'syspaths', 'ThreadPool', 'cprofiles', 'scripts',  
'numProxices', 'componentID', 'componentType', 'uid', 'numClients',  
'globalOrder', 'username', 'load', 'gametime', 'entitiesSize', 'groupOrder')
```

path string(GUIConsolewatcher)

()

```
def scriptLogType( logType ):
```

```
Python.print(: KBEngine.LOG_TYPE_*)
```

def onInit(*isReload*):

([kbengine_defaults.xml](#)->entryScriptFile)

isReload bool

def onFinish():

([kbengine_defaults.xml](#)->entryScriptFile)

bots

bots

PyBots

component

'cell', 'base', 'client', 'database', 'bot' 'editor'

KBEngine


PyClientApp

[KBEngine]

PyClientAppKBEngineC++

```
def getSpaceData( key ):
```

```
def onDestroy( self ):
def onEnterWorld( self ):
def onLeaveWorld( self ):
def onEnterSpace( self ):
def onLeaveSpace( self ):
```



id Integer
entities **Entities**



Entityclient

EntityMAILBOXbasecell .def

def getSpaceData(key):

keyspace
space

setSpaceData

key string

stringkey

def onDestroy(self):

def onEnterWorld(self):

View
cellspace

def onLeaveWorld(self):

View
cellspace

def onEnterSpace(self):

space

def onLeaveSpace(self):

space

entities

entities

Entities

KBEngine

Entity

[KBEEngine]

Entity**KBEEngine** ...

```
import KBEEngine
```

```
def moveToPoint( self, destination, velocity, distance, userData,  
faceMovement, moveVertically ):  
def cancelController( self, controllerID ):
```

```
def onEnterWorld( self ):  
def onLeaveWorld( self ):  
def onEnterSpace( self ):  
def onLeaveSpace( self ):
```

base	MAILBOX
cell	MAILBOX
className	string
clientapp	PyClientApp
direction	Tuple of 3 floats as (roll, pitch, yaw)
id	Integer
position	Vector3
spaceID	uint32
isOnGround	bool



Entityclient

EntityMAILBOXbasecell .def

```
def moveToPoint( self, destination, velocity, distance, userData,
faceMovement, moveVertically ):
```

Entity

ID

```
Entity.cancelController( movementID ) Entity.cancelController(
"Movement" )
```

```
def onMove( self, controllerID, userData ):
def onMoveOver( self, controllerID, userData ):
def onMoveFailure( self, controllerID, userData ):
```

Entity.cancelController

destination Vector3 Entity
velocity float Entitym/s
distance float0
userData object
faceMovement booltruefalse
moveVertically booltruefalse

intID

```
def cancelController( self, controllerID ):
```

cancelController

Entity **real**

controllerID controllerID /entity.cancelController(
"Movement")

def onEnterWorld(*self*):

View
cellspace

def onLeaveWorld(*self*):

View
cellspace

def onEnterSpace(*self*):

space

def onLeaveSpace(*self*):

space

base

base**Base**mailbox **Base**None

Entity.clientEntity
Entity.allClients
Entity.otherClients

MAILBOX

cell

cellcell **MAILBOX**basecellNone

MAILBOX

cellData

cellDatabasecellcell

cell **cellData**cellposition, direction and spaceID

CELLDATADICT

className

string

clientapp

PyClientApp

position

(x, y, z)

Vector3

direction

Entity

Vector3, (roll, pitch, yaw)


isOnGround

True **Entity**False


bool

KBEngine

KBEngineKBEngine



```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
def accountLoginResponse( commitName, realAccountName, extraDatas,
errorCode ):
def createAccountResponse( commitName, realAccountName, extraDatas,
errorCode ):
def chargeResponse( orderID, extraDatas, errorCode ):
def delTimer( id ):
```



```
def onInterfaceAppReady( ):
def onInterfaceAppShutDown( ):
def onRequestCreateAccount( registerName, password, datas ):
def onRequestAccountLogin( loginName, password, datas ):
def onRequestCharge( ordersID, entityDBID, datas ):
```

```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
callbackObj"initialOffset"1"repeatOffset"1
```

```
:
```

```
# addTimer
```

```
import KBEngine
```

```
# 51119
```

```
KBEngine.addTimer( 5, 1, onTimer_Callbackfun )
```

```
# 10
```

```
KBEngine.addTimer( 1, onTimer_Callbackfun )
```

```
def onTimer_Callbackfun( id ):
```

```
print "onTimer_Callbackfun called: id %i" % ( id )
```

```
# if :
```

```
# KBEngine.delTimer( id )
```

initialOffset float

repeatOffset float

callbackObj function

delT

```
:
```

```
integertimeridid delTimer
```

```
def accountLoginResponse( commitName, realAccountName, extraDatas,  
errorCode ):
```

```
onRequestAccountLogin
```

commitName string

realAccountName stringcommitName
extraDatas bytesbasegetClientDatas
integerKBEEngine.SERVER_ERROR_*
errorCode kbengine/kbe/res/server/server_errors.xml
KBEEngine.SERVER_SUCCESS

```
def createAccountResponse( commitName, realAccountName, extraDatas,  
errorCode ):
```

onRequestCreateAccount

commitName string
realAccountName stringcommitName
extraDatas bytesbasegetClientDatas
integerKBEEngine.SERVER_ERROR_*
errorCode kbengine/kbe/res/server/server_errors.xml
KBEEngine.SERVER_SUCCESS

```
def chargeResponse( orderID, extraDatas, errorCode ):
```

onRequestCharge

ordersID uint64ID
extraDatas bytesbasegetClientDatas
integerKBEEngine.SERVER_ERROR_*
errorCode kbengine/kbe/res/server/server_errors.xml
KBEEngine.SERVER_SUCCESS

```
def delTimer( id ):
```

delTimer1delTimer delTimerid

KBEngine.addTimer

id integerid

```
def onInterfaceAppReady( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onInterfaceAppShutDown( ):
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
def onRequestCreateAccount( registerName, password, datas ):
```

```
KBEngine.createAccountResponse
```

```
( kbengine\_defaults.xml->entryScriptFile)
```

```
    registerName string
```

```
    password    string
```

```
    datas      bytes
```

```
def onRequestAccountLogin( loginName, password, datas ):
```

```
KBEngine.accountLoginResponse
```

([kbengine_defaults.xml](#)->entryScriptFile)

loginName string

password string

datas bytes

def onRequestCharge(*ordersID*, *entityDBID*, *datas*):

baseappKBEEngine.charge

KBEEngine.chargeResponse

([kbengine_defaults.xml](#)->entryScriptFile)

ordersID uint64ID

entityDBID uint64DBID

datas bytes


KBEngine

KBEnginePython



```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
def delTimer( id ):
```



```
def onLoggerAppReady( ):
def onLoggerAppShutDown( ):
def onLogWrote( datas ):
def onReadyForShutDown( ):

```

```
def addTimer( initialOffset, repeatOffset=0, callbackObj=None ):
```

```
callbackObj"initialOffset"1"repeatOffset"1
```

```
:
```

```
# addTimer
```

```
import KBEngine
```

```
# 51119
```

```
KBEngine.addTimer( 5, 1, onTimer_Callbackfun )
```

```
# 10
```

```
KBEngine.addTimer( 1, onTimer_Callbackfun )
```

```
def onTimer_Callbackfun( id ):
```

```
print "onTimer_Callbackfun called: id %i" % ( id )
```

```
# if :
```

```
# KBEngine.delTimer( id )
```

initialOffset float

repeatOffset float

callbackObj function

delT

```
:
```

```
integertimeridid delTimer
```

```
def delTimer( id ):
```

```
delTimer1delTimer delTimerid
```

```
KBEngine.addTimer
```

id integerid

```
def onLoggerAppReady( ):
```

```
( kbengine_defaults.xml->entryScriptFile)
```

```
def onLoggerAppShutDown( ):
```

```
( kbengine_defaults.xml->entryScriptFile)
```

```
def onLogWrote( datas ):
```

```
logger
```

```
kbengine_defaults.xml->dbmgr->databaseInterfaces
```

```
( kbengine_defaults.xml->entryScriptFile)
```

```
datas bytes
```

```
def onReadyForShutDown( ):
```

```
( kbengine_defaults.xml->entryScriptFile)
```

boolTrue

KBEngine

Cellapp

CellappBaseappCellappNPC/

KBEngine

Baseapp

Baseapp()

KBEngine