

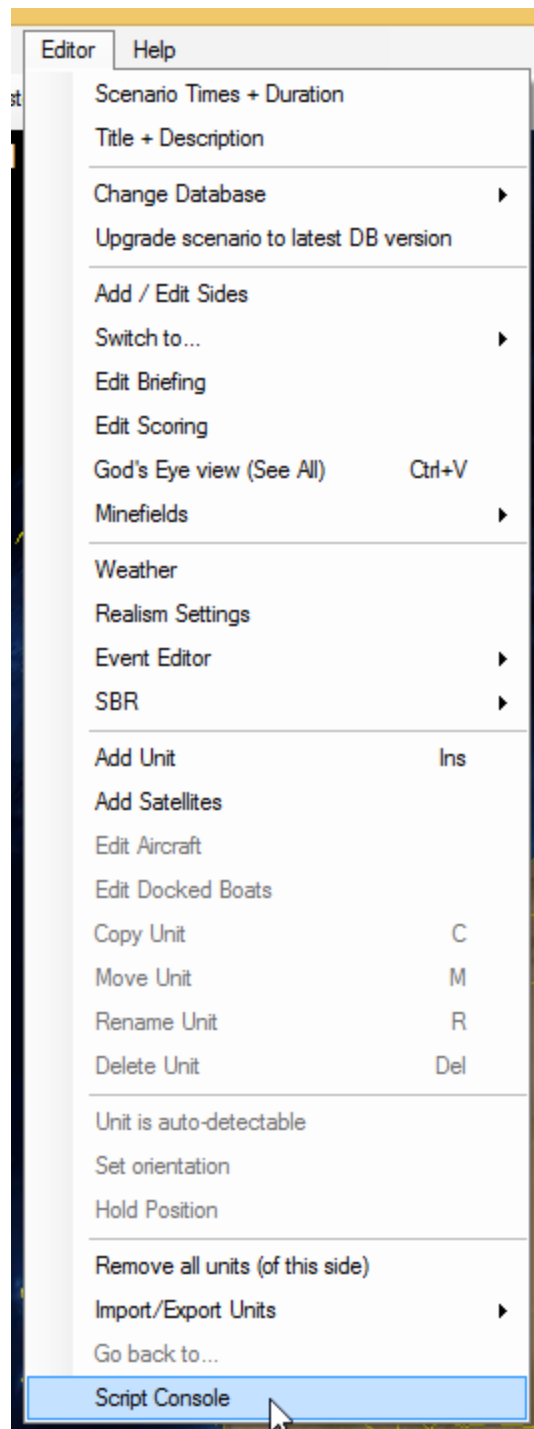
Lua Overview

AND EXAMPLES

Introduction to Lua in Command

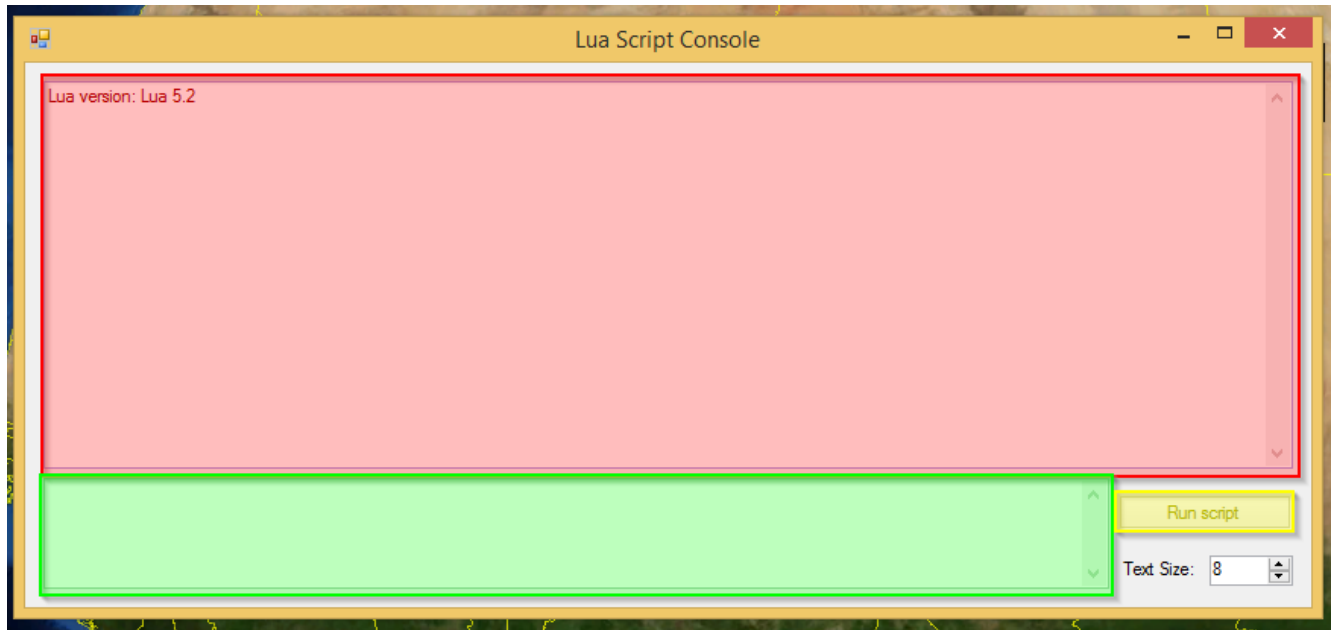
Before you begin, make sure that you've read the first 4 tutorials at the [lua-users wiki](#), to help with understanding what's going on here. We also recommend FunctionsTutorial, but it is optional.

[Scen file, right click -> save as.](#)



Please download the introductory Scene file. This file includes a blank map with two predefined sides, LuaSideA and LuaSideB, who are eternal mortal enemies. These two belligerents want to have a naval incident in the North Atlantic (and each has a mission already set up to do so), but don't have any ships to incident with. We will help them out by giving them some new vessels, using Lua.

Before we get started, we have to introduce our metaphorical office, the place where we'll be working from. In this introduction, we will be using the script console, then in part 2 we will move onto to using Lua from the event editor.



Once you've opened the scene file in the editor, open the script console (Editor->Script Console).

The Script Console provides a place to rapidly create, run, and debug Lua scripts. The interface has 3 components:

- **Red:** This displays the output of your scripts, as well as a history of your inputs.
- **Green:** This is where you enter the scripts into the console to run them.
- **Yellow:** This button is how you execute your script in the game.

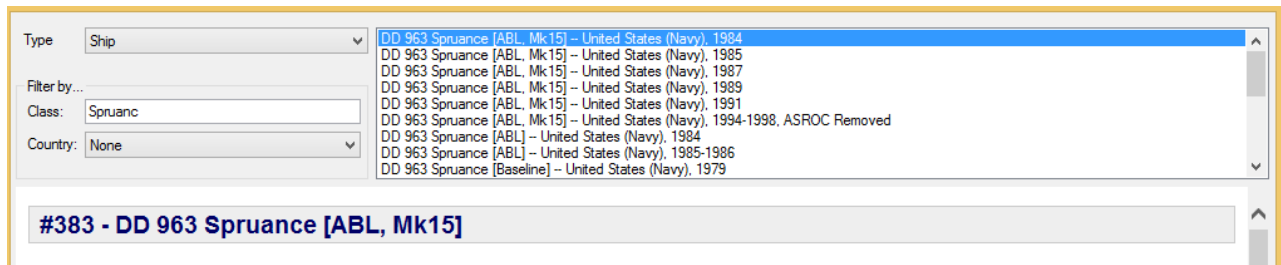
For the purposes of this tutorial, we will be using the script editor to set up a simple scene, where two identical ships fire at each other over a short distance. The scene already has the posture information in it, so we just need to select, position, and create the ships.

Concepts

Before we can get into the meat of Lua in Command, we first have to discuss two basic concepts: IDs and positions.

IDs

Internally, Command refers to every aircraft, ship, submarine, loadout (and rather a lot else) with a unique ID number. To create a unit, you need to tell Command that unit's unique ID number, which can typically be found in the DB viewer.



For the purposes of this tutorial, both LuaSideA and LuaSideB operate the 1984 variant of the Spruance class destroyer. To add it using Lua, we look up the ID in the DB:

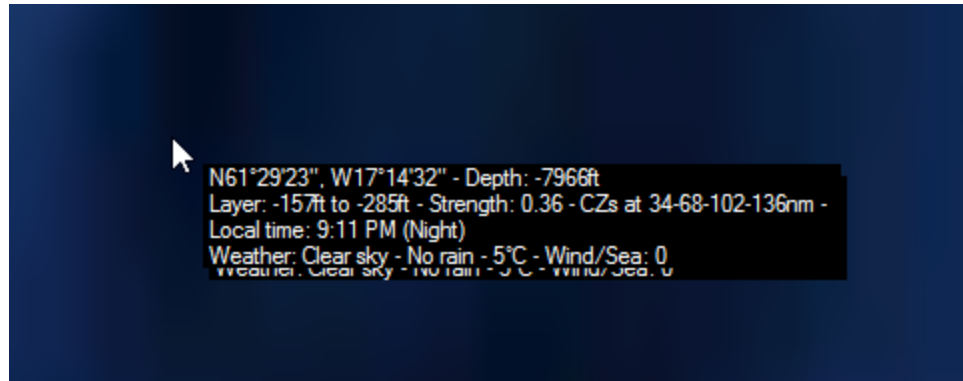
In this case, the class has ID# 383.

Positions

As you probably already know, map positions are referred to with latitude (north/south) and longitude (east/west) coordinates. These can be represented in two forms, degrees minutes seconds, and decimal degrees. The Command Lua API supports both forms, but we will cover only the decimal form.

For this scenario, we will pick an arbitrary location in the North Atlantic for our ships. We pick these two positions:





To copy these positions in-game, point to a position on the map, then press Ctrl-X. This copies a string to the clipboard representing the position pointed to on the map, a string that can then be directly used from Lua.

We will use the following two positions for this tutorial:

- latitude='61.4900913548312', longitude='-17.2428976983679'
- latitude='61.4858175768381', longitude='-17.256394768325'

Note that the formatting is correct for use in a Lua table - this will become important when we come to add them to the scene.

Execution

Let's add our ships now. To do so, we'll need to use the function [ScenEdit AddUnit](#).

[ScenEdit AddUnit](#) adds a unit to the game, using a [unit descriptor](#). A unit descriptor uses a table to define the properties of a unit, such as position, type, and side.

At a minimum, to create a unit, this table needs to contain:

- Side (e.g. "LuaSideA" and "LuaSideB")
- Type (both "Ship" in this case - use the part of the DB you got the ID from)
- Name (in this case "A" and "B")
- DBID (Both 383, a newish Spruance-Class)
- Lat & lon

See the data type documentation for a complete explanation as to the use of these fields.

For our purposes, we want to add two ships with names A and B, DBID 383, at the above coords, to sides LuaSideA and LuaSideB.

```
ScenEdit_AddUnit({side='LuaSideA', type='Ship', name='A', dbid=383,
latitude='61.4900913548312', longitude='-17.2428976983679'})

ScenEdit_AddUnit({side='LuaSideB', type='Ship', name='B', dbid=383,
latitude='61.4858175768381', longitude='-17.256394768325'})
```

Once you run this script, you should end up with two Spruances near Iceland on opposite sides.

Running the scenario now leads to disappointment - the ships don't attack each other. We need to add them to a mission. To do this, we use the [ScenEdit AssignUnitToMission](#) function, with the following signature

```
ScenEdit_AssignUnitToMission('[Unit Name or ID]', '[Mission Name or ID or NONE]')
```

The scene already contains two missions with appropriate setups for attack. We simply need to assign our ships to these missions, like so

```
ScenEdit_AssignUnitToMission('A', 'AStrike')  
ScenEdit_AssignUnitToMission('B', 'BStrike')
```

Now, if you execute this and run the game, the ships should see and attack each other, as intended.

In [tutorial 2](#), we'll cover the basics of aircraft creation and weather alteration.

In this tutorial, we'll discuss how to embed your scripts into a real scenario, as well as how to alter the EMCON of a unit. Yet again, the two sides are fighting, but with ground units which aren't on the map. LuaSideB wants some aerial recon, but can only understand instructions in Lua. This is particularly problematic, because they want a radar sweep but only from within a specific area! Even worse, the generals forgot to tell the pilot about this, so we'll have to inform the pilot to turn the radar on once it arrives.

To implement this challenge, we'll be using Command's events system to control the EMCON of the recon aircraft. The scene file already contains the skeleton of what's needed for Lua use.

Scene file

LuaSideB has already placed the aircraft (BHawk) and the mission (AAnalysis). We'll be configuring BHawk to fly AAnalysis.

SetEMCON

To make this scene work, we'll need to use the [SetEMCON](#) function. This function does what it says on the tin: explains to one of the Luaians what kind of EMCON to use, be it Radar, Sonar, or OECM.

The function's syntax is quite complex. The signature is

```
ScenEdit_SetEMCON(['Side' / 'Mission' / 'Group' / 'Unit'], ['Side  
Name or ID' / 'Mission Name or ID' / 'Group Name or ID' / 'Unit  
Name or ID'], ['Radar/Sonar/OECM=Active/Passive;' / 'Inherit'])
```

For instance, to turn on an BHawk's OECM, you would write

```
ScenEdit_SetEMCON('Unit', 'BHawk', 'OECM=Active')
```

Therefore, we need to do

```
ScenEdit_SetEMCON('Unit', 'BHawk', 'Radar=Active')
```

Events

We need to add Lua to the event framework that's already in the game. We'll add a new action of the type "Lua Action", as seen here:

Event Actions

Description

Create new action: Lua Script Edit selected action Delete selected action

Then, we get

Edit event action

Description:

Settings for action

Add script template: Text Size:

The text entry area is where we'll add our script.

To finish the tutorial, add in our Lua script to set the EMCON, add it to the event, then run the scenario.

Examples:

The lua website with official documentation:
<http://www.lua.org/>

Also michaelm's CMANO Lua reference site:
<http://commandlua.github.io/>

A nice lua IDE (free) is found at
<https://studio.zerobrane.com/>

It contains example code. It will not run Command functions. It will help player learn lua syntax.

Tutorial videos using the zerobrane IDE to help learn lua:
https://www.youtube.com/watch?v=Us46grT9wsA&index=1&list=PL0o3fqwR2CsWg_ockSMN6FActmMOJ70t_

Section edited by Kevin Kinscherf 1-11-2017

```
-----
-----
-----
--
--
Get Latitudes/Longitudes for use in Lua Scripts
--
Written Kevin Kinscherf 12-9-2016
--
Modified NA
--
Modified NA
--
-----
-----
-----
Select a location on the map and hit Cntl-X. This copies the
Latitude/Longitude of the location. This data
can be pasted (Cntl-
V)
into Notepad or a Lua Script. Below is a random sample of a
Cntl-X copy and
paste
.
latitude='8.76340252685485', longitude='109.491225946307'
--
-----
-----
-----
--
--
Get Random Numbers
--
```

Written Kevin Kinscherf 12-9-2016

```
--  
Modified NA
```

```
--  
Modified NA
```

```
-----  
-----  
-----  
Apparently lua is not the best at generating random numbers and r  
returns a lot of
```

```
“
```

```
repeats
```

```
”
```

```
.
```

Here is a work around:

```
math.randomseed(os.time())
```

```
math.random(); math.random(); math.random()
```

```
r = 1
```

```
repeat
```

```
    a = ( math.random( 1, 3 ) )
```

```
    r = r + 1
```

```
until( r > 3 )
```

```
print(a)
```

```
--
```

Written Kevin Kinscherf 12-29-2016

Below is a handy example on how to use the larger data structure called

```
“
```

```
unit
```

```
”
```

. It was provided by Desade

at the Matrix forum:

```
...
```

in your case, it will look like:

```
ScenEdit_AddUnit({type = 'facility', name = 'marine', dbid =
```

```
714, side = 'usa',
```

```
Latitude='9.08471232868106',Longitude='126.147281588214',autodetectable='false',holdfire='false'
```

```
,profi
```

```
ciency=3,mission='base defense'})
```

Few tips:

1. Unit descriptors have lots of parameters, use only those that you need or are needed (for example

AddUnit don't use guid as guid identifies existing unit as an alternative to name/side pair, but when you

add unit there is no guid yet for it). Also important parameter when adding unit is loadout, if unit has them (for example aircraft will need it probably).

2. You can use both " and ' text delimiters, but try not to mix them :)

3. if you are adding lots of units, try to name them differently, for example "marine #1", "marine #2" etc -

exact implementation depends on use case.

4. Due to inheritance by LUA regional settings, above code will fail (unless it's fixed already?) on computers

with comma "," decimal separators due to dot "." in latitude/longitude. Lots of scenarios has this problem, so I found best to use cartographic notation in form of "N 46.00.00"

hope that helps :)

--

Written/Updated by Kevin Kinscherf 1-3-2016

Simple tips used for routine tasks

:

T

o

end your script

pt

and return to CMANO just use the lua statement

“

return

”

(no quotes).

Simple Sub routine

myfunction()

function myfunction()

ScenEdit_MsgBox('In myfunction.', 1)

end

Simple addition Function

function add(first_number, second_number)

print(first_number + second_number)

end

add(100, 3)

Simple Global/Persistent variable

ScenEdit_SetKeyValue("GlobalA", "2")

z = ScenEdit_GetKeyValue("GlobalA")

print(z)

--

Assign Units to Mission

--

Added by Kevin Kinscherf (2016-12-9)

--

NA

--

NA

--

for i=1,4 do

ScenEdit_AssignUnitToMission('Redcock #'..i, "Activate Carrier Gulf")

end

--

unit = "

Redcock

"

```
--
unit
mission = "
Activate Carrier Gulf
.
"
--
mission
--
-----
-----
-----
ScenEdit_AssignUnitToMission (unit, mission)
Both unit and mission are string variables and must exist as nam
ed within the Command scenario bein
g
run. The example also illustrates how to use a FOR loop and st
ring concatenation.
```

```

--
Create reference points around an object on the world map
p
--
Written by Baloogan
--
Modified draw_circle by Yautay
--
Modified by Wayne Stiles (2016-11-14)
--
-----
-----
-----
function draw_circle(a,
b,
p1, pL,
t,
txt)
--
([lat,lon], radius, p1, pL, numpts, name)
  if txt == nil then
    txt =
""
    end
    lat1 = a.latitude
    lon1 = a.longitude
    r = b / 60
  -----
  for i = p1, pL
do
--
Consecutive integer numbers
  th = 2 * math.pi * i / t
  rlat = lat1 + r * math.cos(th)
  rlon = lon1 + r * math.sin(th) / math.cos(math.radians(lat1))
  ScenEdit_AddReferencePoint({
    side = 'PlayerSide',
    lat = rlat,
    lon = rlon,
    name = txt
  })
end
  ..i
,
  highlighted = "yes"})
end
end
--
-----
-----
-----
owner = "
NORAD
"

```

```

--
Side
unit_name = "
Washington D.C
.
"
--
Unit
radius =
4
00
--
Miles from unit
nump =
12
--
Number of points around unit
p1 =
12
--
First number
pL =
23
--
Last number
point_id = "
RP
-
"
--
Reference point ID
--
-----
-----
-----
local unit = ScenEdit_GetUnit({side=owner, name=unit_name
})
draw_circle({latitude=unit.latitude, longitude=unit.longitude}, ra
dius, p1, pL, nump, point_id)

```

This will change the onboard fuel for an aircraft depending upon some condition. In this case if the fuel lever goes below 22,000 lbs it will be reset to the maximum allowed for the aircraft.

To check this level there must be a trigger condition such as one based on time that checks every 15 minutes.

```
u = ScenEdit_GetUnit({side='
USA
',name='
Boeing 707 #1
'}) (from the CWDB database)
local newfuel = u.fuel
if newfuel[2001].current < 22000 then
(
Or some other scenario condition
)
    newfuel[2001].current = newfuel[2001].max
    u.fuel = newfuel
end
```



```

--
Move Aircraft to Base
--
  Added by Kevin Kinscherf (2016-12-9)
--
NA
--
NA
--
-----
-----
-----
AirBase0 = "Aviano"
AirBase1 = "NAS Sigonella"
AirBase2 = "Bezmer AB"
AirBase3 = "Incirlik AB"
r = 1
repeat
  a = ( math.random( 1, 3 ) )
  print(a)
  r = r + 1
if a == 0 then
  for i=1,4 do
    ScenEdit_HostUnitToParent({HostedUnitNameOrID="Warhorse #"..i,
    SelectedHostNameOrID=AirBase0})
  end
end
if a == 1 then
  for i=1,4 do
    ScenEdit_HostUnitToParent({HostedUnitNameOrID="Warhorse #"..i,
    SelectedHostNameOrID=AirBase1})
  end
end
if a == 2 then
  for i=1,4 do
    ScenEdit_HostUnitToParent({HostedUnitNameOrID="Warhorse #"..i,
    SelectedHostNameOrID=AirBase2})
  end
end
if a == 3 then
  for i=1,4 do
    ScenEdit_HostUnitToParent({HostedUnitNameOrID="Warhorse #"..i,
    SelectedHostNameOrID=AirBase3})
  end
end
--
-----
-----
-----
unit = "
Redcock
"
--
unit
mission = "
Activate Carrier Gulf

```

```

.
"
--
mission
--
-----
-----
-----
This code will move 4 aircraft to one of four bases selected a
t random. The print command is only used
for debugging. The code illustrate the FOR loop and string manipulat
on.
ScenEdit_HostUnitToParent (host_info)
host_info:
HostedUnitNameOrID string The name or GUID of the unit to
put into the host
SelectedHostNameOrID string The name or GUID of the host
to put the unit into

```

```

--
Assign Units to 1 of 4 Map Positions
--
Added by Kevin Kinscherf (2016-1-7)
--
NA
--
NA
--
-----
-----
-----
math.randomseed(os.time())
math.random(); math.random(); math.random()
a = math.random(1,4)
if a == 1 then
ScenEdit_AddUnit({type='Sub', side='US', name='SSN 781 USS Californi
a', dbid='550',
latitude='34.2539458437361', longitude='34.1371666398181'})
elseif a == 2 then
ScenEdit_AddUnit({type='Sub', side='US', name='SSN 781 USS Californi
a', dbid='550',
latitude='34.919424257521', longitude='34.423667542601'})
elseif a == 3 then
ScenEdit_AddUnit({type='Sub', side='US', name='SSN 781 USS Californi
a', dbid='550',
latitude='34.5754438255847', longitude='35.0426577712928'})
elseif a == 4 then
ScenEdit_AddUnit({type='Sub', side='US', name='SSN 781 USS Californi
a', dbid='550',
latitude='34.7265771588612', longitude='34.120974472146'})
end
ScenEdit_AssignUnitToMission('SSN 781 USS California', 'Califo
rnia ASW Ptl')
--
-----
-----
-----
unit = "
SSN 781 USS California"
--
unit
mission
=
'California ASW Ptl'
--
mission
--
-----
-----
-----
ScenEdit_AssignUnitToMission (unit, mission)
ScenEdit_AddUnit (unit)
Both unit and mission are string variables and must exist as na
med within the Command scenario being
run. The example also illustrates how to use If logic and string

```

concatenation.

The unit data structure is larger and important to understand.

Source: Live: Old Grudges Never Die

```

-
SetEMCON
--
Written Kevin Kinscherf (2016-12-10)
--
NA
--
NA
--
-----
-----
-----
math.randomseed(os.time())
math.random(); math.random(); math.random()
a = ( math.random( 1, 3 ) )
if a == 1 then
ScenEdit_SetEMCON('Mission', 'RussianSams', 'Radar=Passive')
end
if a == 2 then
ScenEdit_SetEMCON('Mission', 'RussianSams', 'Radar=Active')
end
--
-----
-----
-----
type = "mission"
--
existing mission
name = " RussianSams"
--
name of the mission
emcon
=
'Radar=Active'
--
switch it on/off
--
-----
-----
-----
ScenEdit_SetEMCON (type, name, emcon)

```

Units: Change Side and Assign Mission

Added by Kevin Kinscherf 1-

16

-

17

--

for i=1,16 do

 if ScenEdit_GetUnit({Side='Denmark', Name="Birdsong #"..i,}) ~= nil t

 then

 ScenEdit_SetUnitSide({Side='Denmark', Name="Birdsong #"..i, newside='NATO'})

 if not ScenEdit_GetSidelHuman('NATO') then

 ScenEdit_AssignUnitToMission("Birdsong #"..i, "NATO CAP")

 end

 end

end

Uses four ScenEdit Functions to check if units exist, change

s their side (Denmark to NATO) and

then assigns to an existing NATO mission.

ScenEdit_GetUnit (unit)

Fetches a unit based on a selector.

ScenEdit_SetUnitSide (sidedesc)

Changes the side of a unit

ScenEdit_GetSidelHuman (sidename)

Returns true if side is played by a human player

ScenEdit_AssignUnitToMission (unitname, mission, escort)

Assign a unit to a mission.

Courtesy of LIVE-Brexit