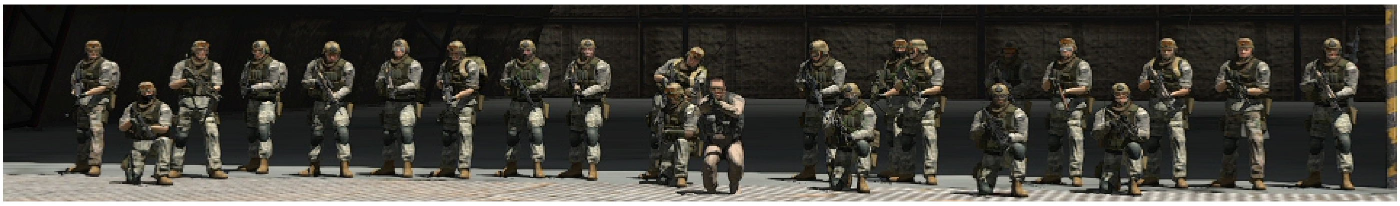


Debugging MOOSE scripts guide - DCS Mission Editing

Author **Message**

Johnny_Rico

offline

Joined: Tue Dec 19, 2017 1:41 pm
Posts: 6

Post subject: Debugging MOOSE scripts guide - DCS Mission Editing **Posted:** Sat Sep 29, 2018 11:37 am

I hope this quick guide can help you out when debugging MOOSE scripts in DCS World
It is only about 4 pages long and will give you valuable info on how to go about solving problems
BUT more importantly it will show you how to get the info you will need to get help from people on the MOOSE discord channel
https://discord.gg/dZ6xAr
The MOOSE discord channel is full of people who can help you out ..... but **YOU need to put in a little effort first!**
Just posting in discord "this does not work" .... without giving dcs.log with your code ..... is not going to get you much help

1. Make sure you are using the latest MOOSE DEVELOPMENT version

URL for static MOOSE develop branch -> https://github.com/FlightControl-Master/MOOSE_INCLUDE/tree/develop/Moose_Include_Static

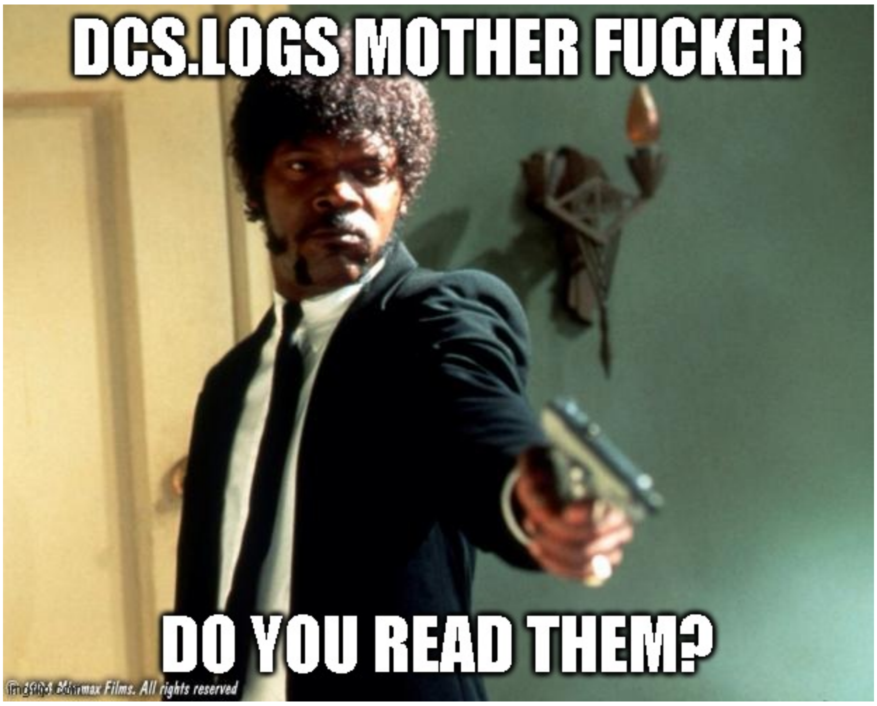
URL Latest demo missions - very useful to learn from -> https://github.com/FlightControl-Master/MOOSE_MISSIONS/tree/develop

2. Get some tools to help you out.

Text editor -> Notepad++ - https://notepad-plus-plus.org/

Log viewer -> hoo wintail - http://www.hootech.com/WinTail/
Log viewer -> glogg - https://glogg.bonnefon.org/
Log viewer -> Baretail - https://www.baremetalsoft.com/baretail/

3. Jules wants you to get to know your dcs.log

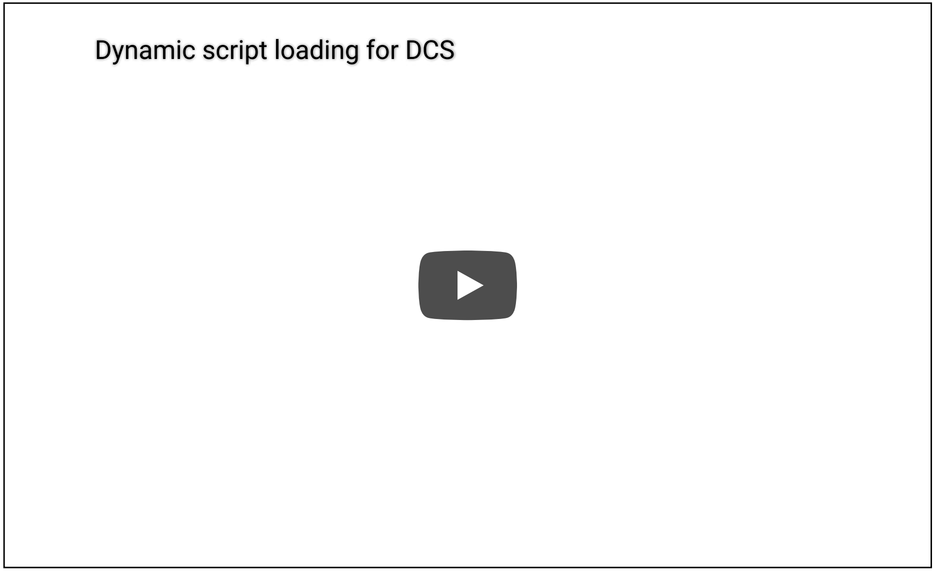


DCS.log which is found in **C:\Users\USERNAME\Saved Games\DCS\Logs** or

C:\Users\USERNAME\Saved Games\DCS.openbeta\Logs

USERNAME = your windows username you are logged in under
dcs.log is one file you will be wanting to monitor to keep an eye on what your scripts are doing, use one of the above log viewer tools to do this. When something is not working right checking the log can help you debug it
DCS.log contains a running log of what DCS.exe is doing, when things happen they are added to this log
when things go wrong with scripts they are also added to this log file so it is a useful debugging aid!

4. Learn how to load scripts dynamically (BIGGEST TIME SAVER)



This will make you debugging faster and easier to do eg you want to test a new function
this will save you a lot of time - Big shout to out to Pikster for creating this video - I owe you a lot of hours of my life i got back

5. RTFM - Read the fine Manual

If in doubt about something go have a look at the MOOSE docs
New MOOSE docs -> https://flightcontrol-master.github.io/MOOSE_DOCS_DEVELOP/Documentation/
Then go have a look at some of the demo missions
here -> https://github.com/FlightControl-Master/MOOSE_MISSIONS/tree/develop
Play the mission, open the mission , examine and have a look at the script inside and learn
Hoggit has a Wiki of DCS scripting functions -> https://wiki.hoggitworld.com/view/Category:Functions#
MOOSE Docs tips:
You can use **control + F** - and enter a term to search for on a Doc page, will speed up things for you

6. Watch some Youtube tutorial videos

Flightcontrol has made a lot of good tutorial videos talking you through some of the features
that MOOSE has, these are great for the more "visual" type learners out there
His video channel -> https://www.youtube.com/channel/UCjrA9j5LQoWsG4SpS8i79Qg

7. Use debug output statements to help see where you code flow is going:

Code:

```
number_of_bugs_in_DCS = 5
function  test_function() -- this is defining the function itself

   local value = false

   if number_of_bugs_in_DCS < 1 then
      value = true
      env.info(" ++++++++ Debug Output - Bugs = Zero - This is either a miracle or you are high")
   elseif number_of_bugs_in_DCS < 10 then
      value = false
      env.info(" ++++++++ Debug Output - Bugs < 10 - some text")
   end

   return value
end -- EOF

test_function() -- this is the call to the above function
```

This simple example above shows how to use simple debug output to the dcs.log file to show which parts
of the code are actually being executed when it is run.

8. Some Common things to look out for

A) Is moose loading correctly ? Examine dcs.log (near the start of when you ran your mission) for a line that looksa bit like this:
***** MOOSE GITHUB Commit Hash ID:** 2020-03-20T15:37:50.0000000Z-9351b269e2afa3024bb0224ce3b534fbeedd089b ***
and bit lower down
SCRIPTING: * MOOSE INCLUDE END *****
If those a present then MOOSE has loaded ok (remember to load MOOSE before any scripts that rely on it)

B) Are you loading MOOSE before your scripts?
1st trigger set to type 4 MISSION START conditions (leave empty) and ACTIONS use this to load moose.lua
2nd trigger set to type 1 ONCE conditions TIME MORE (1) ACTIONS use this to load your scripts you want to use in the mission

C) nil errors .... what are they?
when lua encounters something it does not know about it throw **nil error** this can be:
objects eg 1. you have tried to spawn an object (tank for example) and you try to access it in some way before it was ready
2. you have a spawn template in the ME that you want to use called "Plane"
but when you tried to use it for spawning you used my_spawn = SPAWN:New("**p**lane")

methods eg trying to call and method to access an ME placed zone called "capture"
myzone = **Zone**:FindByName("capture")
things are case sensitive in lua !
myzone = **ZONE**:FindByName("capture")
The second one above is using the correct spelling, checking the docs or using intellisence will help you out

Parameters eg trying to send a parameter to a function that is not the accepted type - send a string, when it expects a number visa versa
If a method is showing a nil error, check the spelling of the function call, if that is correct check the docs to make sure you are sending a valid parameter

chaining methods (compound execution) eg trying to do multiple things together
position3d = GROUP:FindByName("plane"):GetVec3()
what you are trying to do is find a group named "plane" then return its Vec3 (its position in 3d space) to variable position3d
BUT, what happens if "plane" does not exist right now when this is called ? a **nil error**!
A lot of the time if you call a function and it "fails" then its return value will be nil, MOOSE will then try and call a method on a nil and fail
so better way is

Code:

```
mygroup = GROUP:FindByName( "plane")
if mygroup ~= nil then
 position3d = mygroup:GetVec3()
end
```

9. Use debug output to work out what data is inside an object

(what it is made up of) Sometimes you might not be sure of what data items are contained in an object,
so it would be nice to be able to peek inside. I use this simple functions to do this for me

Code:

```
-- use this to out co-ordinates
function debug_cords(place)
  env.info("------------------------  Vec3.y = " .. place.y .. "  Vec.x = " .. place.x .. " Vec.z = " ..
place.z )
end

-- if you do not know what type of data object you are dealing with use this function to dump it out to dcs.log
function debug_object(object, name)
   local result_table = routines.utils.oneLineSerialize(object)
   if name == nil then
      name = ""
   end
   env.info("  " )
   env.info("-------------- DEBUG OBJECT START " .. name .. " ----- DEBUG OBJECT START " ..name.. " ----- DEBUG
OBJECT START "..name.. " --------------")
   env.info( result_table )
   env.info("-------------- DEBUG OBJECT END " .. name .. " ----- DEBUG OBJECT END " ..name.. " ----- DEBUG
OBJECT END "..name.. " --------------")
   env.info("  ")
end -- EOF

-- if you know it is an array table use this function (name is used to make output pretty and ID it)
function debug_array( t_table, name )
   if name == nil then
      name = ""
   end
   env.info("  " )
   env.info("-+-+-+-+-+-+-+ DEBUG ARRAY START " .. name .. " +-+-+ DEBUG ARRAY START " ..name.. " +-+-+ DEBUG
ARRAY START "..name.. " +-+-+-+-+-+-+-")
   for i, item in ipairs(t_table) do
      env.info(" -+-+ Index " .. tostring(i) .. " - " .. tostring(item))
   end
   env.info("-+-+-+-+-+-+-+ DEBUG ARRAY END " .. name .. " +-+-+ DEBUG ARRAY END " ..name.. " +-+-+ DEBUG ARRAY
END "..name.. " +-+-+-+-+-+-+-")
   env.info("  " )
end -- EOF
```

example call:
debug_object(test_zone, " TEST ZONE ") -- send a zone object into the function and print out out " TEST ZONE " is just so delimiter text
There is more info on the built in debug output from MOOSE itself here -> https://flightcontrol-master.github.io/MOOSE_DOCS/Documentation/Core.Base.html
Delta99 Will be writing full guide to help users use this shortly (he is an expert in this)

10. Example script mistake

- mission started but things did not work right - a Menu was missing
A) pause the mission - lets not create anymore log data to trawl through than we need to
B) lets see what the dcs.log has to say - start from top and work downwards until you see errors

log shows this

```
2018-09-29 11:16:10.279 INFO SCRIPTING: 11654( 12320)/E: SET_GROUP02692._FilterStart({[1]=Adding Object:,[2]=Transport
Helicopter,})
2018-09-29 11:16:10.280 ERROR DCS: Mission script error: : [string
"C:\Users\Matej\AppData\Local\Temp\DCS.openbeta\/~mis0000182B"]:35287: attempt to index field 'CargoGroup' (a nil value)
stack traceback:
[C]: ?
[string "C:\Users\Matej\AppData\Local\Temp\DCS.openbeta\/~mis0000182B"]:35287: in function 'New'
[string "C:\Users\Matej\AppData\Local\Temp\DCS.openbeta\/~mis00004AA2"]:37: in main chunk
2018-09-29 11:16:10.281 INFO SCRIPTING: 11654( 12320)/E: SET_GROUP02704._FilterStart({[1]=Adding Object:,[2]=AWACS,})
2018-09-29 11:16:10.281 INFO SCRIPTING: 11654( 12320)/E: SET_GROUP02704._FilterStart({[1]=Adding Object:,[2]=EWR,})
```

To work out WTF has gone wrong we work BACKWARDS from the bottom upwards of the error
[string "C:\Users\Matej\AppData\Local\Temp\DCS.openbeta\/~mis00004AA2"]:37: in main chunk
^ this is telling us that on line 37 of our script something has went wrong

So we go to line 37 which is
local AttackCargoAlpha = CARGO_GROUP:New(GROUP:FindByName("Attack"), "Attack", "Assault Team Alpha", 500):RespawnOnDestroyed(false)

and try and work out which of the things we are sending to the CARGO_GROUP:New function thing are wrong. So we examine the code
above this paying attention to the spelling / case of the items defined. If that seems ok then time to look at the ME
Did I name the things Iam trying to reference in the script correectly ? no typos or EXTRA spaces ? (not it can be hard to spot an extra
space in the DCS ME - so take care)

11. Some Final Words

Be AWARE, if you are developing scripts for a Multiplayer mission ....
When you test stuff out in SP it may NOT run the same in multiplayer!!!!!!!!
eg
You test the mission in the mission editor SP and the scripts all run fine and things work as expected
You load the mission and SELF HOST it in MP and thing "seem" to work fine
You then load the mission on a dedicated MP server and it does NOT work correctly??????????

SP != MP self hosting != MP dedicated server

Bugs in how DCS runs in MP may cause all sorts of ballache, best try and test out sooner rather than later before you waste too much time

Develop Environment

if you are more adventurous you can grab the source and set-up a dev env
Eclipse - https://www.eclipse.org/ldt/
Visual Code - https://code.visualstudio.com/download
Github for downloading Repo -> https://desktop.github.com/
Github URL for Develop branch -> https://github.com/FlightControl-Master/MOOSE_INCLUDE.git
Webpage for develop branch -> https://github.com/FlightControl-Master/MOOSE_INCLUDE/tree/develop
In eclipse you can use Control + H, to do a search for info on functions etc

Top profile

Display posts from previous: All posts Sort by Post time Ascending Go

newtopic postreply Page **1** of **1** [1 post]

Board index » Public Section » Virtual Air Force All times are UTC

Who is online

Users browsing this forum: No registered users and 1 guest

Jump to: Virtual Air Force Go