CommonAPI2 Remote Script Debugging

Inhaltsverzeichnis

- 1 Einführung
- 2 Auswahl der IDE / Debugger
  - 2.1 ZeroBrane Studio
  - 2.2 Visual Studio Code
    - 2.2.1 devcat Lua Debugger
  - 2.3 Probleme beim schreiben in Globals
  - 2.4 Lua Panda

1 Einführung

Seit CommonAPI2 Version 202003025 Windows ist es möglich via luasocket lua Scripte zu "remote" debuggen.

D.h. man kann Breakpoints setzen und sich dann die Variablen Inhalte anschauen bzw. sein Script verfolgen.

Wichtiger Hinweis:

TPF2 und CommonAPI2 haben mehrere zu großen Teilen unabhängige LUA States.

Alle externen Debugger kommen nur jeweils mit einer LUA Umgebung (State) zurecht.

2 Auswahl der IDE / Debugger

Es gibt mehrere IDEs und Debugger.

- Alle externen Debugger kommen nur jeweils mit einer LUA Umgebung (State) zurecht.
- Der Debugger muss via luasocket (socket.core) kommunizieren
- luasocket muss in den CommonAPI2 Einstellungen aktiviert sein. (Speichern nicht vergessen und spiel neu starten)

2.1 ZeroBrane Studio

https://studio.zerobrane.com/

(Es funktioniert die Portable Version)

Ihr benötigt hierfür https://github.com/pkulchenko/…b/master/src/mobdebug.lua ,

diese Datei (der eigentliche Debugger) müsst Ihr in `eis_os_commonapi_2/res/scripts` kopieren.

In ZeroBrane Studio:

Wählt unter Project -> Project Directory -> Choose das Verzeichnis eures Mods: meinmod_1

Unter Project -> Start Debug Server einschalten.

In einer updateFn oder in anderen Teilen des Codes, der den Debugger starten soll:

```
require("mobdebug").start()
```

Sobald TPF über diesen Code stolpert, wird es den Debugger laden und bei Breakpoints in ZeroBrane Studio anhalten.

Damit .con Dateien auch als lua Dateien erkannt werden:

Edit -> Settings Users

Die Zeile `editor.specmap.con = 'lua'` hinzufügen und speichern, ein Neustart der IDE wird benötigt.

2.2 Visual Studio Code

Es gibt mehrer Plugins

2.2.1 devcat Lua Debugger

In VSCode installieren:

https://marketplace.visualstud…itemName=devCAT.lua-debug

Ihr benötigt hierfür https://github.com/devcat-stud…uggee/vscode-debuggee.lua

diese Datei (der eigentliche Debugger) müsst Ihr in `eis_os_commonapi_2/res/scripts` kopieren.

Damit es keine Fehler in VS Code gibt, die Datei mit Breakpoint würde nicht existieren, in der vscode-debuggee.lua:

sourceBasePath = initMessage.sourceBasePath

[2] https://dev.transportfever.net/lexicon/entry/299-commonapi2-remote-script-debugging/

ersetzen durch

sourceBasePath = commonapi._native.paths.getGamePath()

Beispiel einer launch.json für VS Studio:

Code

```
{

        {









        }
    ]
}
```

Alles anzeigen

Da CommonAPI2 schon dkjson mitliefert, kann man den Debugger in einem LUA Script so starten:

Code

```
local         json         =         require         'commonapi2.extlib.dkjson'
setmetatable(_G,                                {}                                )
local          debuggee          =          require          'vscode-debuggee'
debuggee.start(json)
```

2.3 Probleme beim schreiben in Globals

Damit man Globals schreiben kann, kann vorher setmetatable(_G, {}) nutzen.

2.4 Lua Panda

https://marketplace.visualstud…mName=stuartwang.luapanda

Ungetestet, ein Test von Enzojz sieht vielversprechend aus.