Updated: 06/29/25

Script Launcher

This program is meant to be a supplement to the AmazonReader project that runs a script as defined by the specifications for that project. This runs as a separate entity that communicates with AmazonReader when it is run with the -network option. This causes AmazonReader to wait for commands to be sent from the network on a TCP port. The interface between the programs uses TCP protocol and the AmazonReader can be set to a specified port selection. It will initially set this value to the port selected by the user the last time the program was run (it is saved in the file `.amazonreader/site.properties` in the directory that ScriptLauncher is executed from. If this file is not found, it will default to port 6000. The same way, ScriptLauncher has a file .scriptlauncher/site.properties that is used to hold the last port used by this program, and the startup screen allows the user to specify another value to override this value.

CLIENT Messages Sent

The commands (ASCII text) that are sent from ScriptLauncher to the AmazonReader server are:

```
LOAD <filename>  - loads a script into memory
COMPILE          - compile the script
RUN              - execute the script
STOP             - stop the script (if it is running)
PAUSE            - pause the execution (so it can be resumed)
RESUME           - resume execution from the current point
STEP             - execute the next line in the script
RESET            - reset the program counter (next line to execute) to the begining
BREAKPT <line>   - set a breakpoint in the script to stop execution when running
DISCONNECT       - disconnect the TCP port from the server
EXIT             - terminate the server
```

SERVER Responses Received

The responses sent back from the server are:

```
STATUS: <state>         - a change of state in the server
LINE: <line>            - the current line number
SUBSTACK: <stack info>  - the current list of subroutine calls
ALLOC: <variable info>  - each variable allocation created during compile
VARMSG: <variable info> - variables that have changed during script execution
LOGMSG: <log info>      - debug log info (info the server also saves in log file)
OUTPUT: <message>       - output generated by the script using the PRINT command
```

STATUS Response Format

```
CONNECTED        - the client has connected to the server
DISCONNECTED     - the client has disconnected from the server
LOADED           - the script file has completed loading
COMPILED         - the script has completed compilation
EOF              - the execution of the script has completed
STOPPED          - the execution has stopped due to client request
PAUSED           - the execution has paused due to client request
STEPPED          - the execution has completed for the current line
BREAK            - the execution has paused due to reaching the set breakpoint
RESUMED          - the execution has resumed
BREAKPT SET      - the breakpoint change has been accepted
BREAKPT INVALID  - the breakpoint change was rejected (not a valid exacutable line)
```

This status value will be displayed in the Status text message location.

LINE Response Format

The LINE entry is simple enough: it will be the line number of the script that is the next to be executed. If the end of the script has been reached, the value will be `END`.

SUBSTACK Response Format

```
SUBSTACK: MAIN :: Subroutine1 :: Subroutine2
```

The current subroutine will be the furthest name on the right, and the caller of each subroutine listed preceeds it with a double colon seperating them. The leftmost entry will always be MAIN. This information is displayed in the Sub stack text message location.

ALLOC Response Format

```
ALLOC: [<name> VarName :: <type> String :: <owner> *MAIN*]
```

The entries are again seperated by a double colon and consist of an id tag followed by its corresponding value. The tags are: <name>, <type>, and <owner>, although <owner> will be omitted for the RESERVED section. <name> refers to the name of the variable, <type> is the data type, and <owner> is either MAIN or the subroutine in which the allocation was made. This is more important in LOCAL section variables, since they are only accessible to that subroutine. This information is formatted and displayed in the Variables tab.

VARMSG Response Format

This message is similar to ALLOC, but has additional fields. It has the following format:

```
VARMSG: [<section> RESERVED :: <name> VarName :: <type> Integer :: <value> 32 :: <writer>
        FileStatusCheck :: <line> 166 :: <time> 00:00.000]
```

The additional sections here are <section>, to identify whether it is a RESERVED, GLOBAL, LOCAL or LOOP variable; <value> which is the current value of the variable, <writer> is the subroutine that wrote the value, <line> is the line in the script that wrote the variable, and <time> is the execution time when the value was written (same values that are used in the log messages, for reference). This information is also formatted and displayed in the Variables tab, with any variable changes indicated by marking the values in red.

LOGMSG Response Format

LOGMSG info consists of the debug information that is saved in the AmazonReader test log file. It has the format of:

```
LOGMSG: 00000664 00:00.062 [PROG  ]:     Path selection: /home/user/Projects/AmazonLogger/logs
```

Where the 8-digit entry is a counter value (for reference), followed by the timestamp value in msec resolution as MM:SS.mmm, followed by the type, of debug message enclosed in square brackets, followed by the message itself. This information is displayed in the Debug log tab.

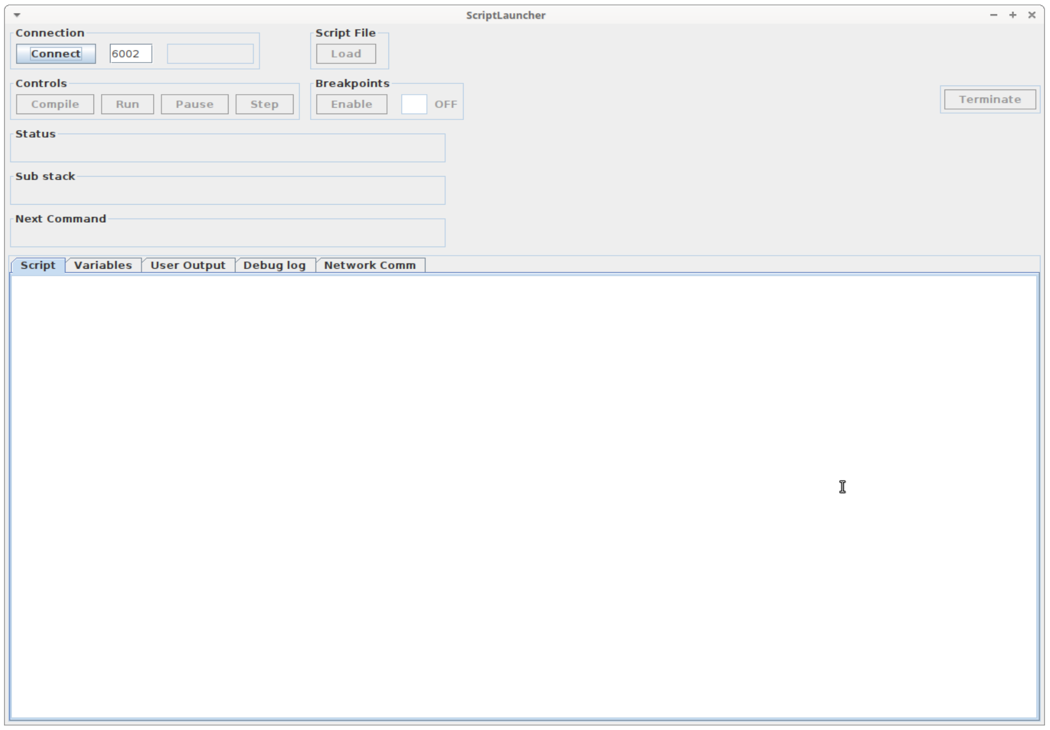
OUTPUT Response Format

OUTPUT info is simply the text output generated by the PRINT messages in the script. This information is displayed in the User Output tab.

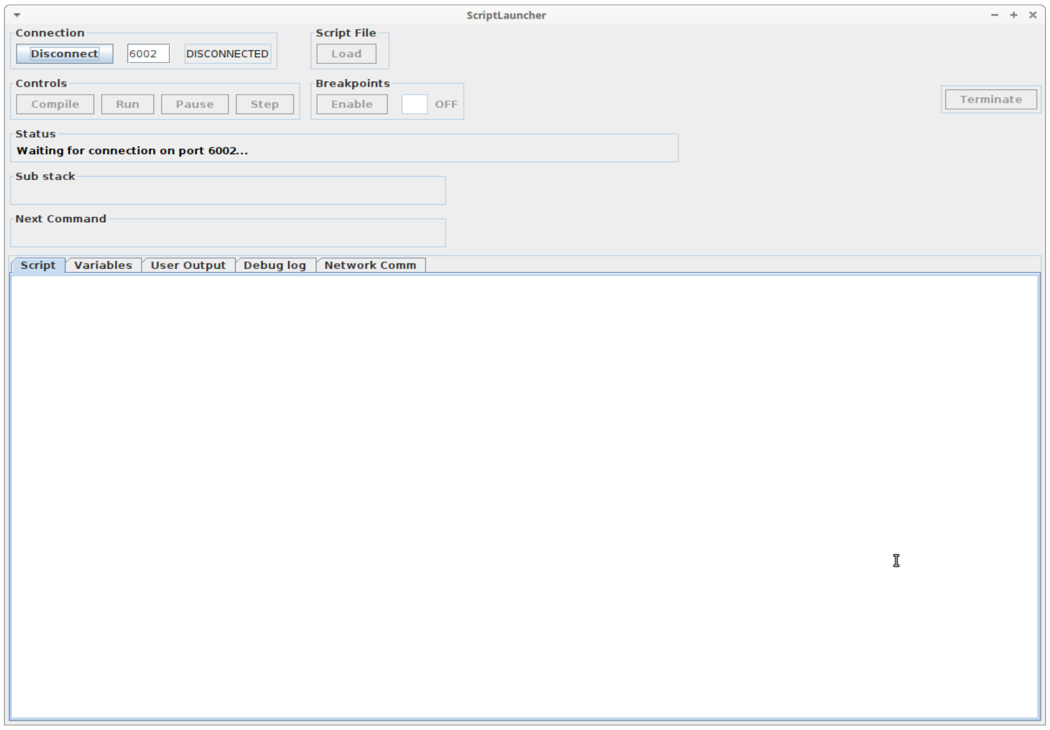
Note that all communication between the client and server are also displayed in the Network Comm tab with client messages displayed in blue and server messages in green. Additional status info is displayed in black.

Startup

This is the GUI that is displayed upon startup:



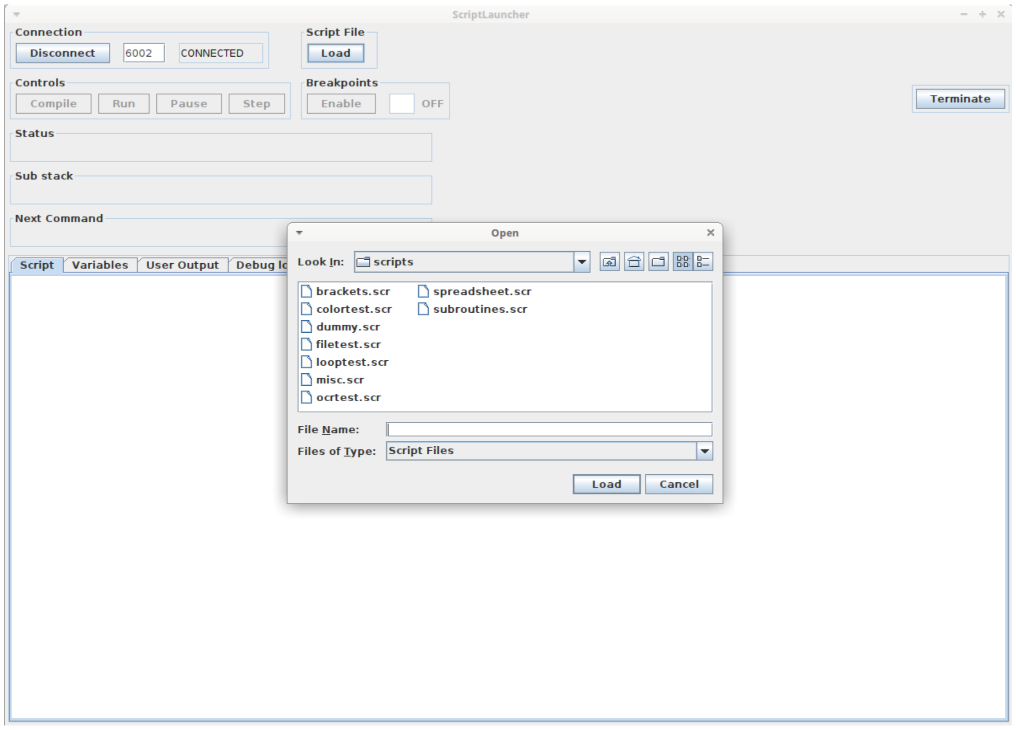
Note that all the buttons are disabled except for the Connect on powerup, so you must first connect to the server before doing anything else. The AmazonReader must be running with the -network option in order to connect. If not, the Status display will indicate it is waiting for the server and will connect after AmazonReader server is started.



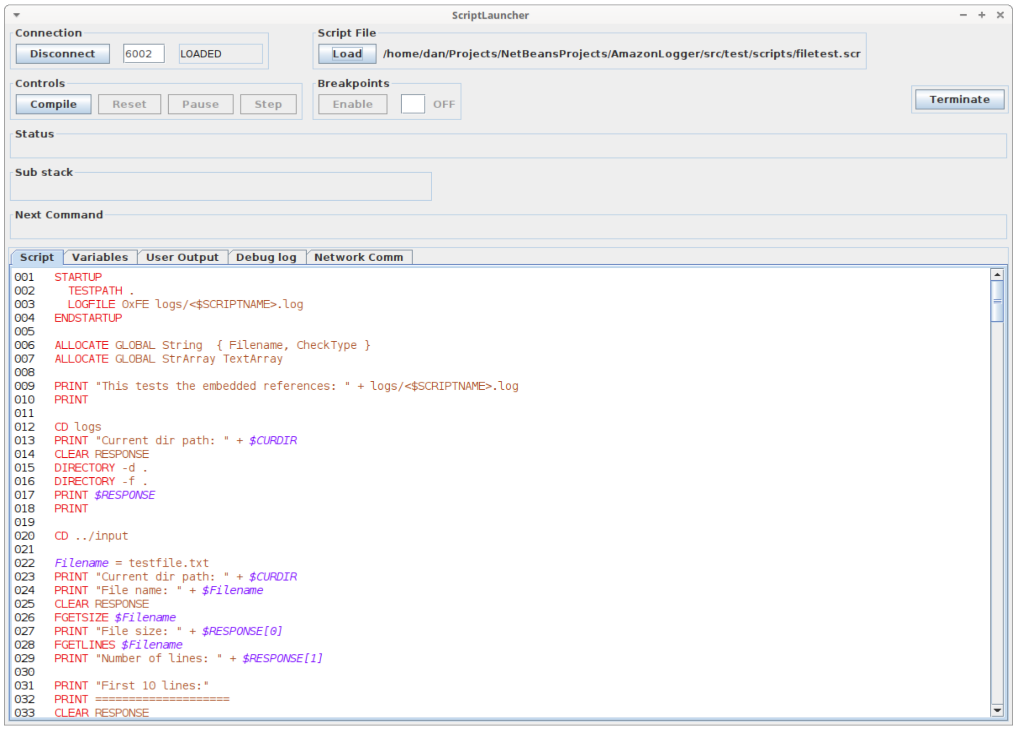
Waiting for connection on port 6002...

When the server accepts a connection, it will send the client a CONNECTED status message to this client, and the Connect button will change to a Disconnect button and the Load button will become available to press.

Pressing the Load button allows you to select a test script to run. The filename will be passed to the server in the LOAD command, and the server will load the file and return a LOADED response when complete.

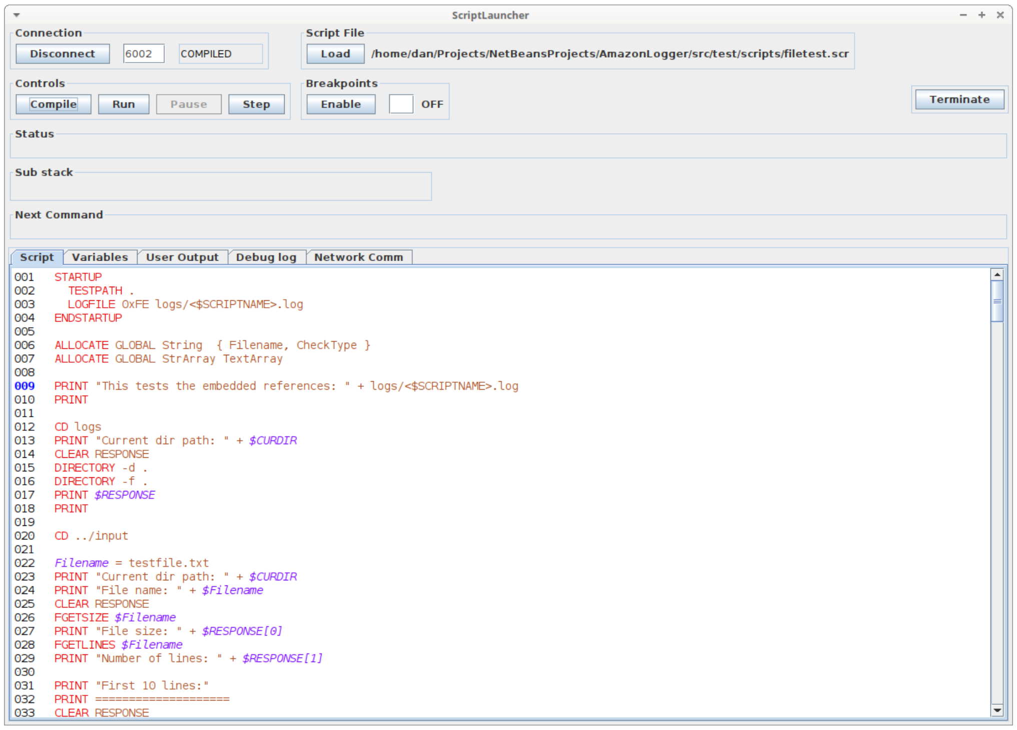


After a file has been loaded, the loaded file will be displayed in the Script tab and the Compile button becomes available to press. Line numbers are displayed at the left of each line for reference. The window will show script commands in red, variable references in violet, comments in green and the rest of the script in brown.



```
001   STARTUP
002     TESTPATH .
003     LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004   ENDSTARTUP
005
006   ALLOCATE GLOBAL String  { Filename, CheckType }
007   ALLOCATE GLOBAL StrArray TextArray
008
009   PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010   PRINT
011
012   CD logs
013   PRINT "Current dir path: " + $CURDIR
014   CLEAR RESPONSE
015   DIRECTORY -d .
016   DIRECTORY -f .
017   PRINT $RESPONSE
018   PRINT
019
020   CD ../input
021
022   Filename = testfile.txt
023   PRINT "Current dir path: " + $CURDIR
024   PRINT "File name: " + $Filename
025   CLEAR RESPONSE
026   FGETSIZE $Filename
027   PRINT "File size: " + $RESPONSE[0]
028   FGETLINES $Filename
029   PRINT "Number of lines: " + $RESPONSE[1]
030
031   PRINT "First 10 lines:"
032   PRINT ====================
033   CLEAR RESPONSE
```

Pressing the Compile button will send the COMPILE command to the server, which will compile the script, sending LOGMSG messages that show its progress (note that the desired debug messages to report must be enabled at the start of the script in order to obtain reports). These log messages will always include error and warning messages, whether enabled or not, which can allow the user to debug the script to get it to at least compile. During the compile is when all variable allocations that were made in the script are allocated. These will be sent to the client using the ALLOC response messages. A COMPILED message will be sent upon completion, which will enable the Run, Step, and Breakpoint Enable buttons.



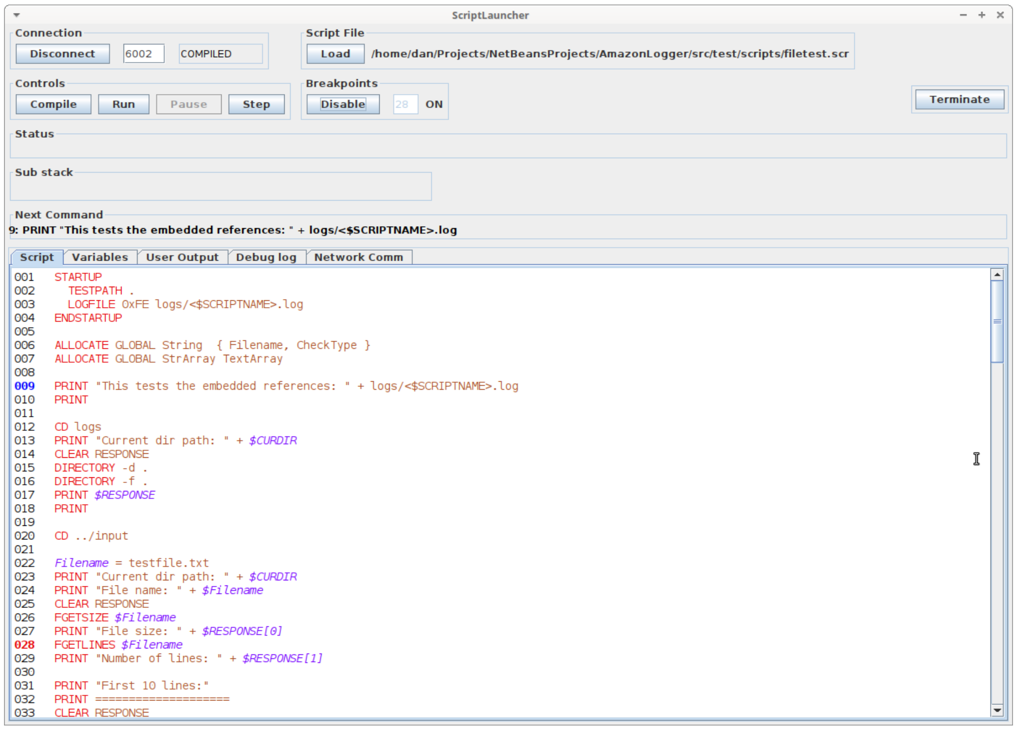
Connection: 6002 COMPILED

Script File: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

Breakpoints: OFF

```
001  STARTUP
002    TESTPATH .
003    LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004  ENDSTARTUP
005
006  ALLOCATE GLOBAL String  { Filename, CheckType }
007  ALLOCATE GLOBAL StrArray TextArray
008
009  PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010  PRINT
011
012  CD logs
013  PRINT "Current dir path: " + $CURDIR
014  CLEAR RESPONSE
015  DIRECTORY -d .
016  DIRECTORY -f .
017  PRINT $RESPONSE
018  PRINT
019
020  CD ../input
021
022  Filename = testfile.txt
023  PRINT "Current dir path: " + $CURDIR
024  PRINT "File name: " + $Filename
025  CLEAR RESPONSE
026  FGETSIZE $Filename
027  PRINT "File size: " + $RESPONSE[0]
028  FGETLINES $Filename
029  PRINT "Number of lines: " + $RESPONSE[1]
030
031  PRINT "First 10 lines:"
032  PRINT ===================
033  CLEAR RESPONSE
```

The line number of the initial executable line will be displayed in blue to indicate what line will be executed next. Note that the commands contained in the STARTUP section and all ALLOC commands will be skipped, as these have been executed by the compiler. Also, all blank and comment lines will also be skipped. Here you can enter a line number to break on.



Connection: 6002 COMPILED

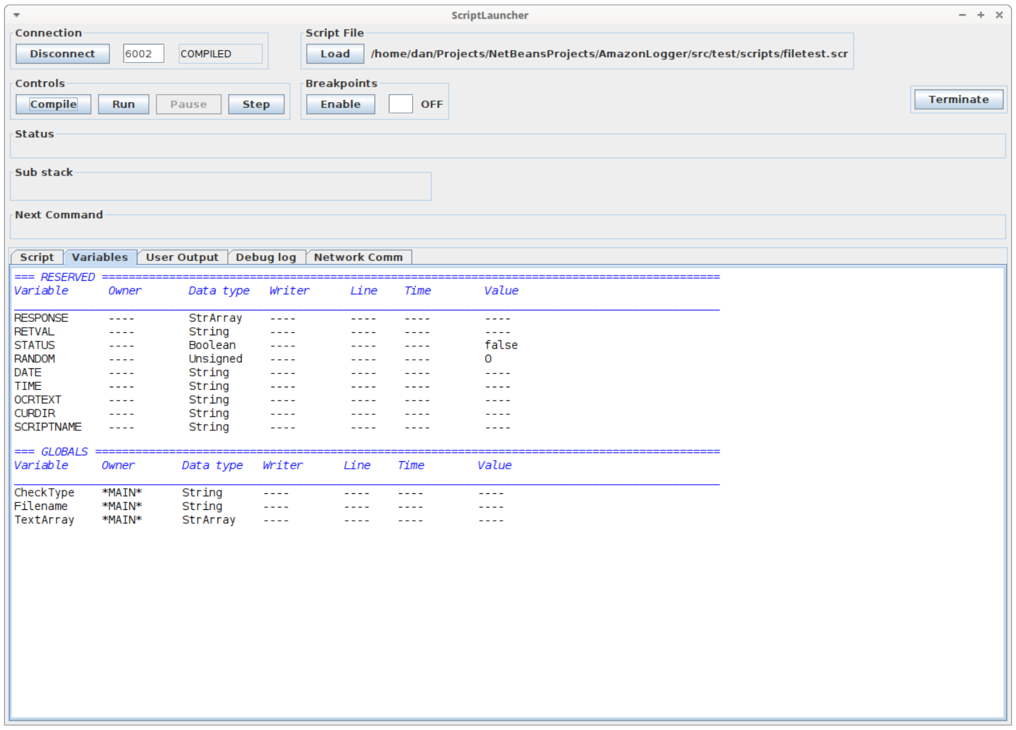
Script File: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

Breakpoints: 28 ON

Next Command: 9: PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log

```
001  STARTUP
002    TESTPATH .
003    LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004  ENDSTARTUP
005
006  ALLOCATE GLOBAL String  { Filename, CheckType }
007  ALLOCATE GLOBAL StrArray TextArray
008
009  PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010  PRINT
011
012  CD logs
013  PRINT "Current dir path: " + $CURDIR
014  CLEAR RESPONSE
015  DIRECTORY -d .
016  DIRECTORY -f .
017  PRINT $RESPONSE
018  PRINT
019
020  CD ../input
021
022  Filename = testfile.txt
023  PRINT "Current dir path: " + $CURDIR
024  PRINT "File name: " + $Filename
025  CLEAR RESPONSE
026  FGETSIZE $Filename
027  PRINT "File size: " + $RESPONSE[0]
028  FGETLINES $Filename
029  PRINT "Number of lines: " + $RESPONSE[1]
030
031  PRINT "First 10 lines:"
032  PRINT ===================
033  CLEAR RESPONSE
```

Note that the breakpoint line number will be displayed in red, so you can easily see where the script will stop execution at.



6002 COMPILED

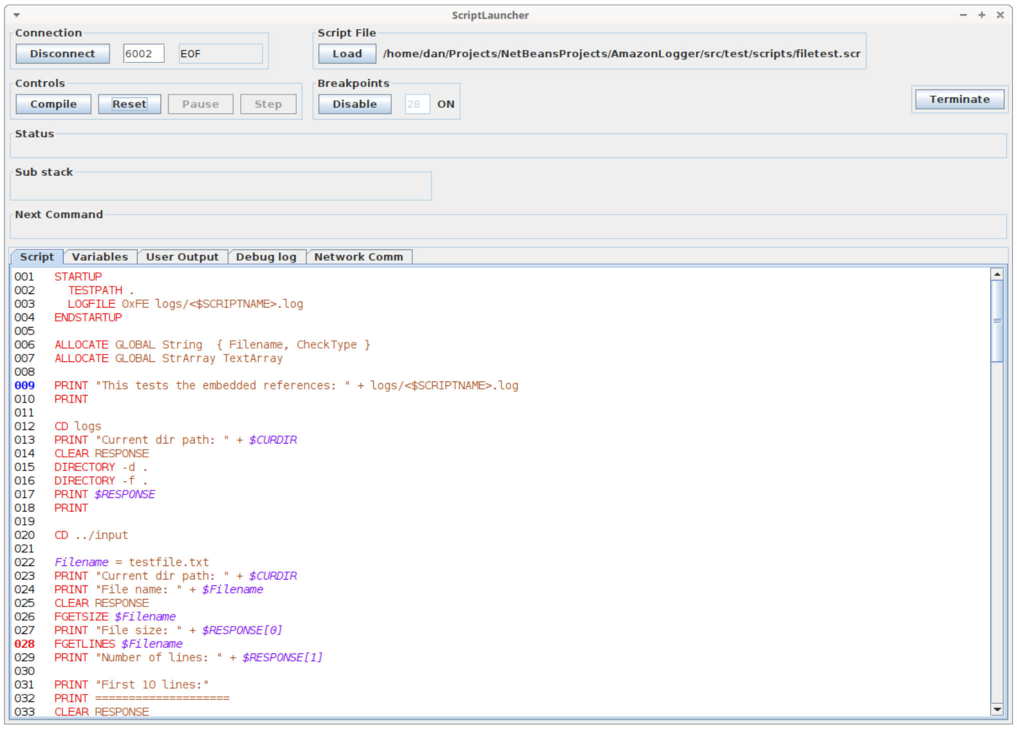
/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

OFF

```
=== RESERVED ==================================================================================
Variable      Owner       Data type  Writer      Line    Time        Value
________________________________________________________________________________________________
RESPONSE      ----        StrArray   ----        ----    ----        ----
RETVAL        ----        String     ----        ----    ----        ----
STATUS        ----        Boolean    ----        ----    ----        false
RANDOM        ----        Unsigned   ----        ----    ----        0
DATE          ----        String     ----        ----    ----        ----
TIME          ----        String     ----        ----    ----        ----
OCRTEXT       ----        String     ----        ----    ----        ----
CURDIR        ----        String     ----        ----    ----        ----
SCRIPTNAME    ----        String     ----        ----    ----        ----

=== GLOBALS ====================================================================================
Variable     Owner       Data type  Writer      Line    Time        Value
________________________________________________________________________________________________
CheckType    *MAIN*      String     ----        ----    ----        ----
Filename     *MAIN*      String     ----        ----    ----        ----
TextArray    *MAIN*      StrArray   ----        ----    ----        ----
```

If you look at the Variables tab, you will see all of the Reserved and User defined allocations that the compiler found and their data types, the subroutine (or MAIN) that created them, and their default values if Boolean, Integer, or Unsigned types. All Strings and Arrays default to empty.



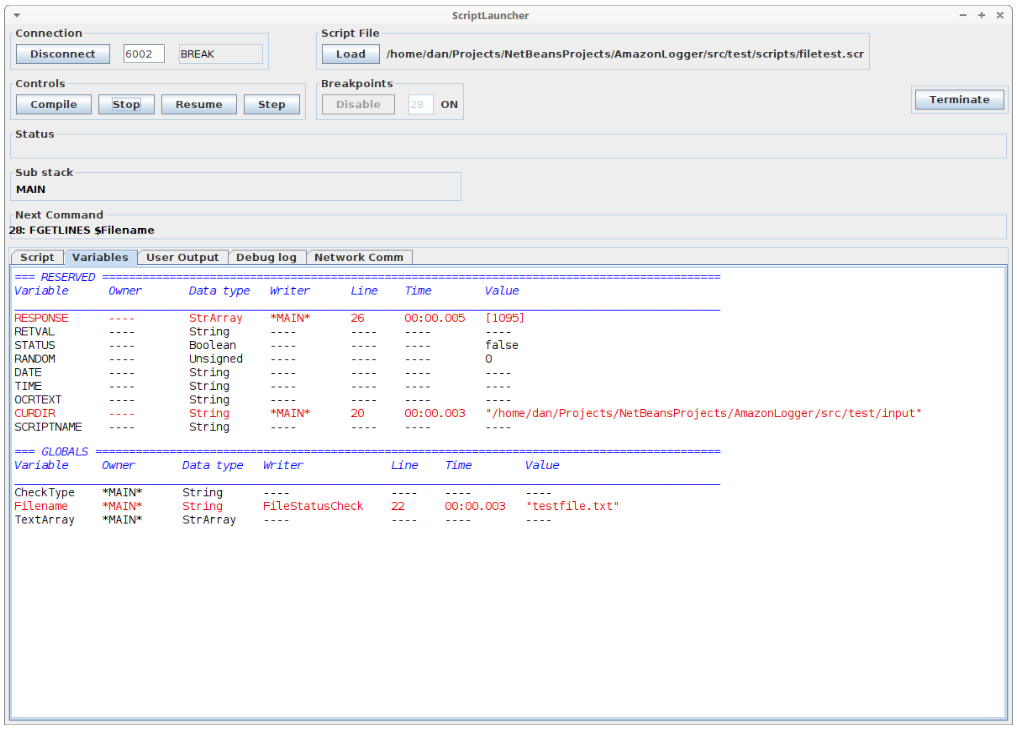
6002 EOF

/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

28 ON

```
001   STARTUP
002     TESTPATH .
003     LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004   ENDSTARTUP
005
006   ALLOCATE GLOBAL String  { Filename, CheckType }
007   ALLOCATE GLOBAL StrArray TextArray
008
009   PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010   PRINT
011
012   CD logs
013   PRINT "Current dir path: " + $CURDIR
014   CLEAR RESPONSE
015   DIRECTORY -d .
016   DIRECTORY -f .
017   PRINT $RESPONSE
018   PRINT
019
020   CD ../input
021
022   Filename = testfile.txt
023   PRINT "Current dir path: " + $CURDIR
024   PRINT "File name: " + $Filename
025   CLEAR RESPONSE
026   FGETSIZE $Filename
027   PRINT "File size: " + $RESPONSE[0]
028   FGETLINES $Filename
029   PRINT "Number of lines: " + $RESPONSE[1]
030
031   PRINT "First 10 lines:"
032   PRINT ===================
033   CLEAR RESPONSE
```

If the Run button is pressed, it will execute the command in the script until either the end of the file is reached, the user presses the STOP or PAUSE buttons, or the enabled breakpoint is reached. The above shows that the breakpoint was reached and indicates it by updating the line numbers to show that the breakpoint line number of 28 is now violet, as well as displaying in the text area after the port number that a BREAK Status message was received from the server. Looking at the Variables tab, we can see the variables that have changed in value highlighted in red.

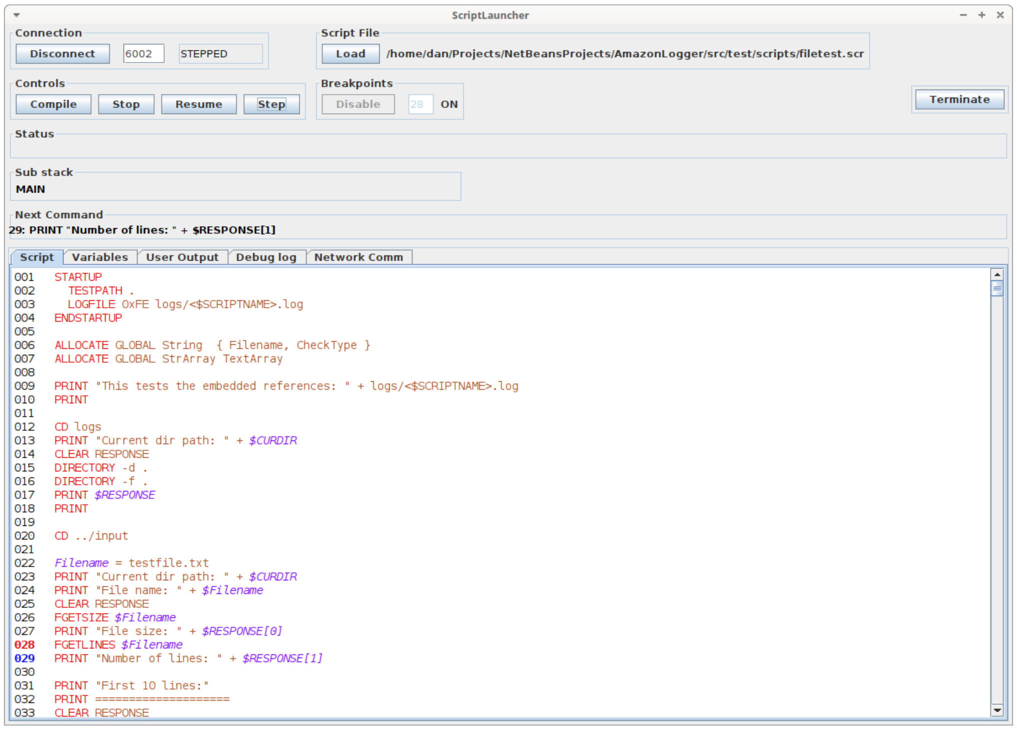


```
=== RESERVED ==========================================================================================
Variable      Owner       Data type   Writer      Line    Time        Value
______________________________________________________________________________________________________
RESPONSE      ----        StrArray    *MAIN*      26      00:00.005   [1095]
RETVAL        ----        String      ----        ----    ----        ----
STATUS        ----        Boolean     ----        ----    ----        false
RANDOM        ----        Unsigned    ----        ----    ----        0
DATE          ----        String      ----        ----    ----        ----
TIME          ----        String      ----        ----    ----        ----
OCRTEXT       ----        String      ----        ----    ----        ----
CURDIR        ----        String      *MAIN*      20      00:00.003   "/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input"
SCRIPTNAME    ----        String      ----        ----    ----        ----

=== GLOBALS ===========================================================================================
Variable     Owner       Data type   Writer            Line    Time        Value
______________________________________________________________________________________________________
CheckType    *MAIN*      String      ----              ----    ----        ----
Filename     *MAIN*      String      FileStatusCheck   22      00:00.003   "testfile.txt"
TextArray    *MAIN*      StrArray    ----              ----    ----        ----
```

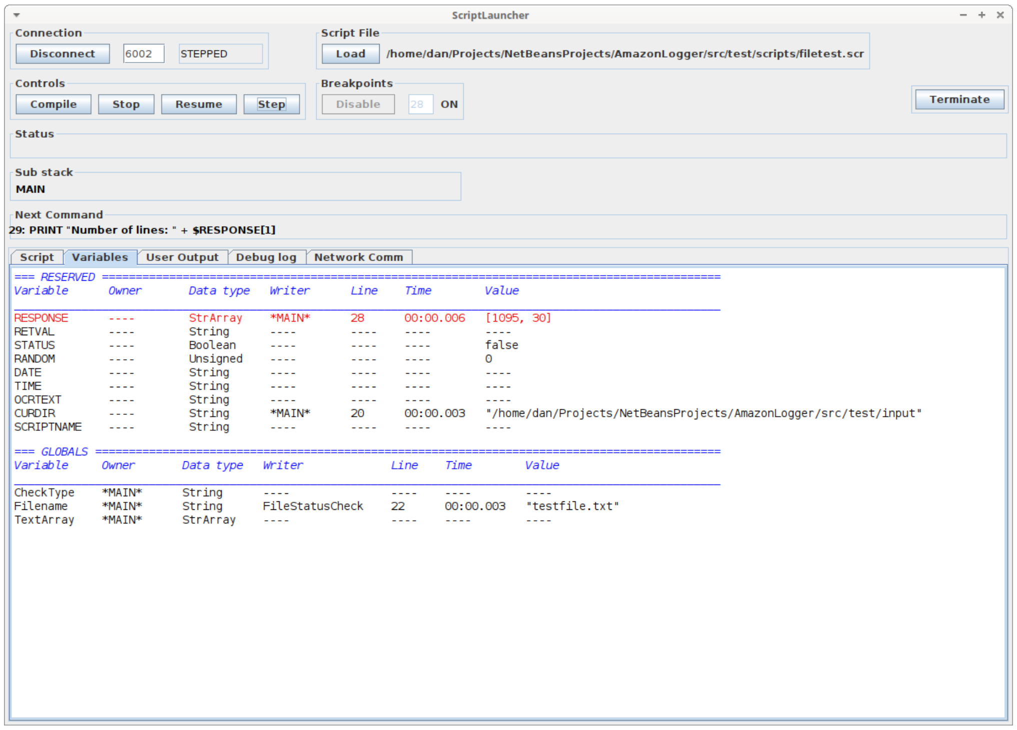
This shows that the RESPONSE string array has been written to by the MAIN routine on line 26 at timestamp 0.005 (5 msec after start of execution). This was from the `FGETSIZE $Filename` line executed. This is a String Array, and values are shown enclosed in square brackets with each entry seperated by a comma. In this case, there was only 1 value written, and it was a numeric value of 1095.

Taking a step after the execution break, the next slide again displays the breakpoint in red on line 28, but the next line to be executed is again indicated in blue at line 29.



```
001   STARTUP
002     TESTPATH .
003     LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004   ENDSTARTUP
005
006   ALLOCATE GLOBAL String  { Filename, CheckType }
007   ALLOCATE GLOBAL StrArray TextArray
008
009   PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010   PRINT
011
012   CD logs
013   PRINT "Current dir path: " + $CURDIR
014   CLEAR RESPONSE
015   DIRECTORY -d .
016   DIRECTORY -f .
017   PRINT $RESPONSE
018   PRINT
019
020   CD ../input
021
022   Filename = testfile.txt
023   PRINT "Current dir path: " + $CURDIR
024   PRINT "File name: " + $Filename
025   CLEAR RESPONSE
026   FGETSIZE $Filename
027   PRINT "File size: " + $RESPONSE[0]
028   FGETLINES $Filename
029   PRINT "Number of lines: " + $RESPONSE[1]
030
031   PRINT "First 10 lines:"
032   PRINT ==================
033   CLEAR RESPONSE
```

We can again select the Variables tab to see the changes made.



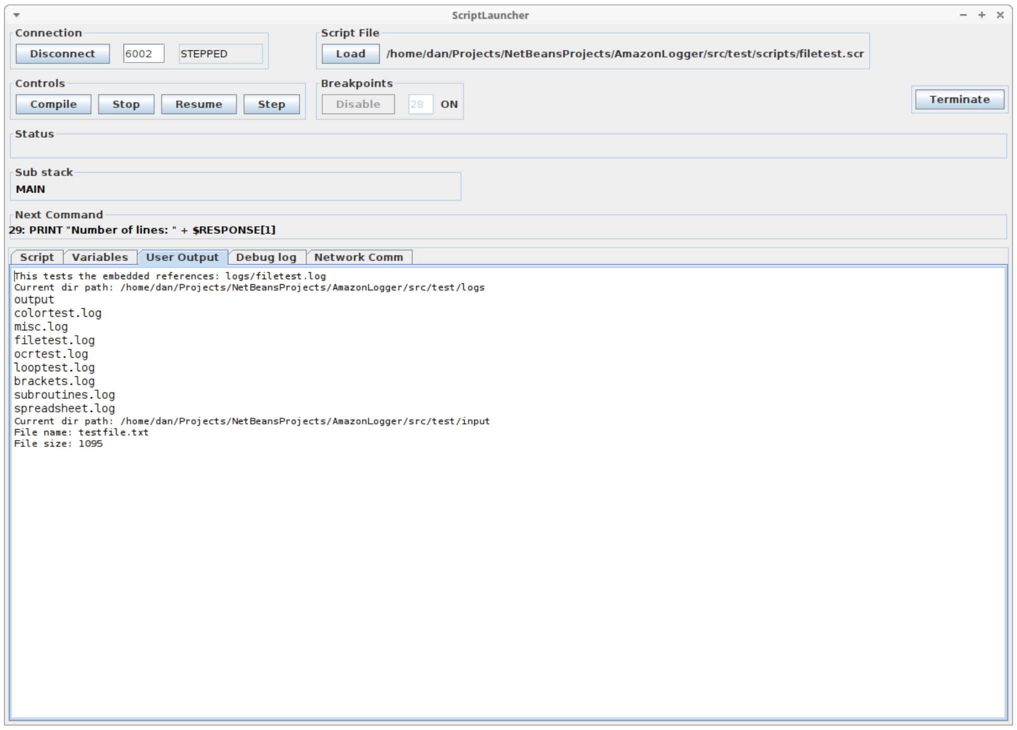
```
29: PRINT "Number of lines: " + $RESPONSE[1]
```

```
=== RESERVED ===================================================================================
Variable      Owner        Data type   Writer      Line    Time         Value
________________________________________________________________________________________________
RESPONSE      ----         StrArray    *MAIN*      28      00:00.006    [1095, 30]
RETVAL        ----         String      ----        ----    ----         ----
STATUS        ----         Boolean     ----        ----    ----         false
RANDOM        ----         Unsigned    ----        ----    ----         0
DATE          ----         String      ----        ----    ----         ----
TIME          ----         String      ----        ----    ----         ----
OCRTEXT       ----         String      ----        ----    ----         ----
CURDIR        ----         String      *MAIN*      20      00:00.003    "/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input"
SCRIPTNAME    ----         String      ----        ----    ----         ----

=== GLOBALS ====================================================================================
Variable     Owner        Data type   Writer            Line    Time         Value
________________________________________________________________________________________________
CheckType    *MAIN*       String      ----              ----    ----         ----
Filename     *MAIN*       String      FileStatusCheck   22      00:00.003    "testfile.txt"
TextArray    *MAIN*       StrArray    ----              ----    ----         ----
```

It shows that the variable RESPONSE has been written to by MAIN again line 28 at timestamp 0.006, and now the array contains 2 entries: 1095 and 30. This was performed by line 28 that was `FGETLINES $Filename`.

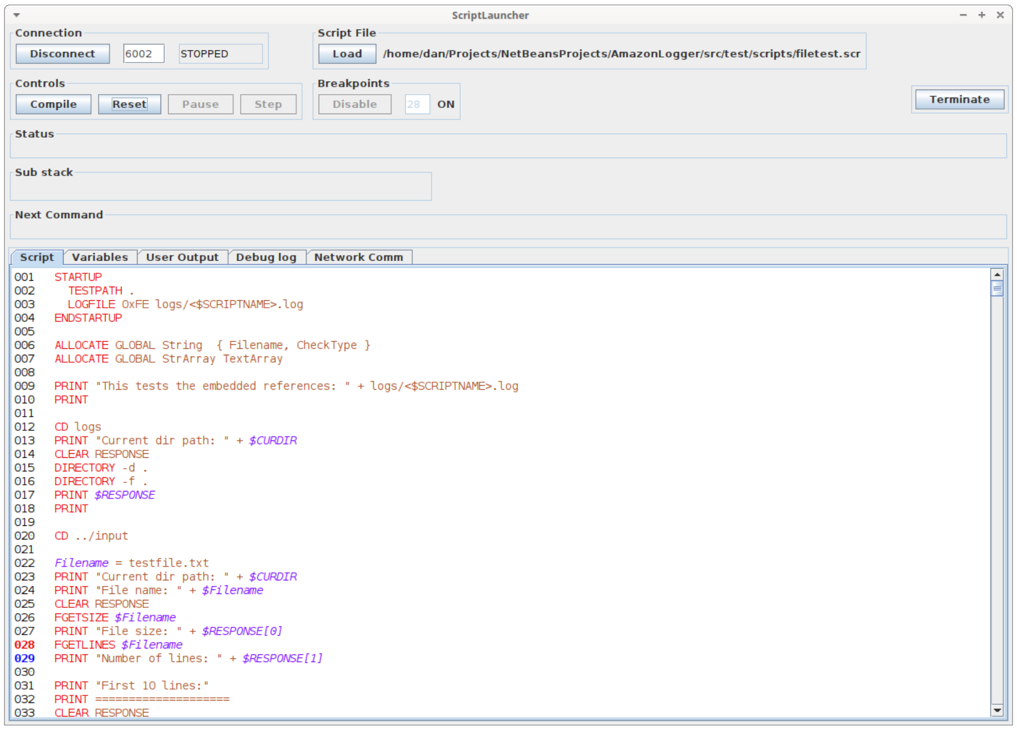
Next we can look at the Output tab to see what the PRINT messages have written.



```
29: PRINT "Number of lines: " + $RESPONSE[1]
```

```
This tests the embedded references: logs/filetest.log
Current dir path: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs
output
colortest.log
misc.log
filetest.log
ocrtest.log
looptest.log
brackets.log
subroutines.log
spreadsheet.log
Current dir path: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input
File name: testfile.txt
File size: 1095
```

We can stop the program by pressing the STOP button, which will terminate the script.



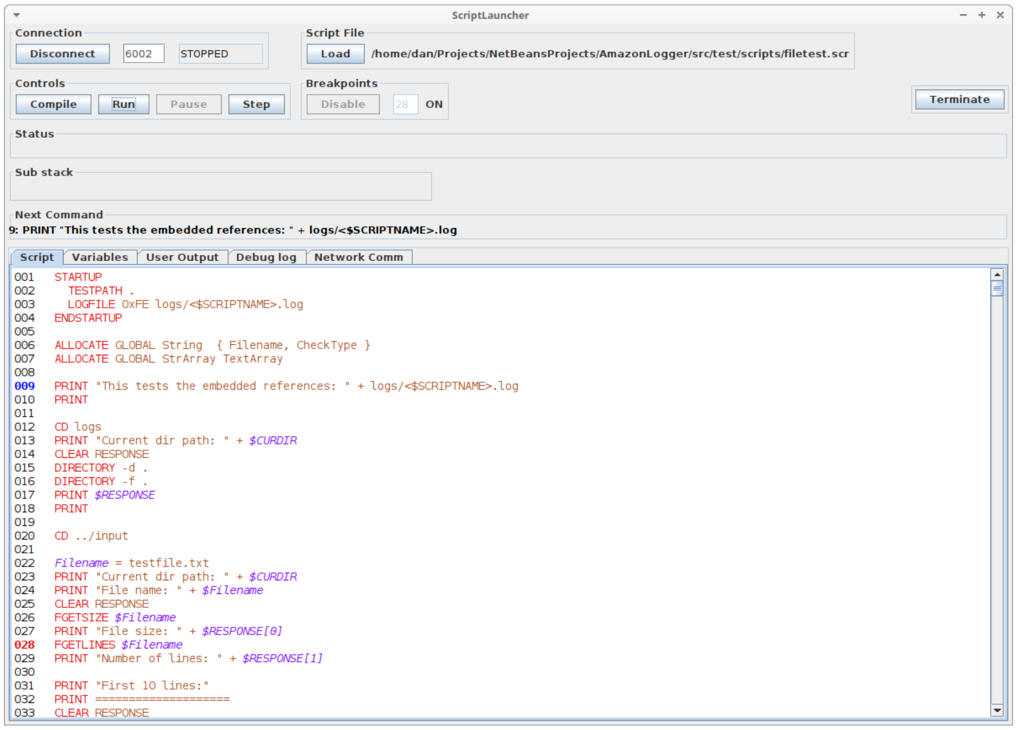
Connection: 6002 STOPPED

Script File: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

Breakpoints: 28 ON

```
001  STARTUP
002    TESTPATH .
003    LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004  ENDSTARTUP
005
006  ALLOCATE GLOBAL String  { Filename, CheckType }
007  ALLOCATE GLOBAL StrArray TextArray
008
009  PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010  PRINT
011
012  CD logs
013  PRINT "Current dir path: " + $CURDIR
014  CLEAR RESPONSE
015  DIRECTORY -d .
016  DIRECTORY -f .
017  PRINT $RESPONSE
018  PRINT
019
020  CD ../input
021
022  Filename = testfile.txt
023  PRINT "Current dir path: " + $CURDIR
024  PRINT "File name: " + $Filename
025  CLEAR RESPONSE
026  FGETSIZE $Filename
027  PRINT "File size: " + $RESPONSE[0]
028  FGETLINES $Filename
029  PRINT "Number of lines: " + $RESPONSE[1]
030
031  PRINT "First 10 lines:"
032  PRINT ===================
033  CLEAR RESPONSE
```

And if we want to run the script again, we must first press the RESET button.



Connection: 6002 STOPPED

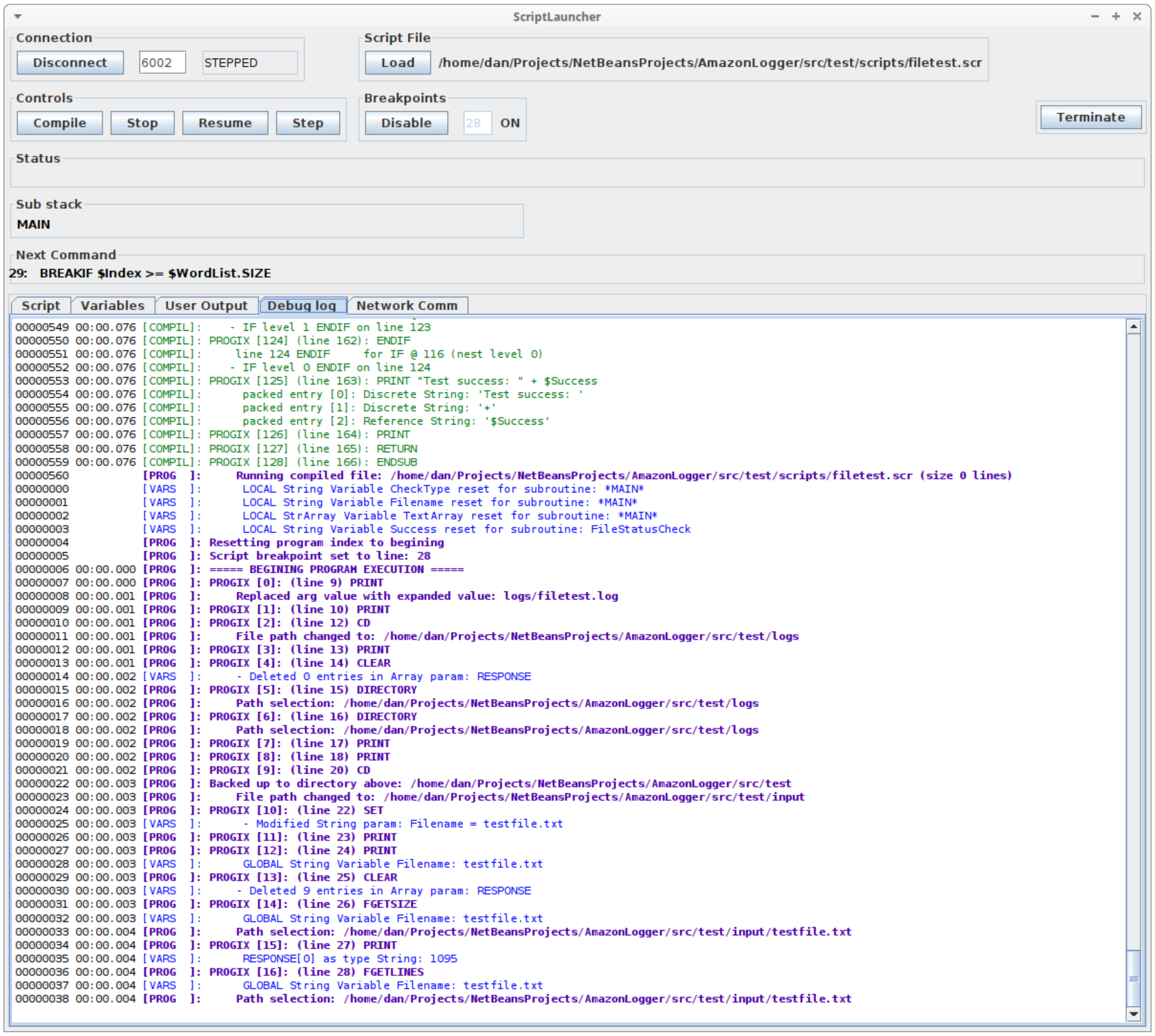
Script File: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr

Breakpoints: 28 ON

Next Command: 9: PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log

```
001  STARTUP
002    TESTPATH .
003    LOGFILE 0xFE logs/<$SCRIPTNAME>.log
004  ENDSTARTUP
005
006  ALLOCATE GLOBAL String  { Filename, CheckType }
007  ALLOCATE GLOBAL StrArray TextArray
008
009  PRINT "This tests the embedded references: " + logs/<$SCRIPTNAME>.log
010  PRINT
011
012  CD logs
013  PRINT "Current dir path: " + $CURDIR
014  CLEAR RESPONSE
015  DIRECTORY -d .
016  DIRECTORY -f .
017  PRINT $RESPONSE
018  PRINT
019
020  CD ../input
021
022  Filename = testfile.txt
023  PRINT "Current dir path: " + $CURDIR
024  PRINT "File name: " + $Filename
025  CLEAR RESPONSE
026  FGETSIZE $Filename
027  PRINT "File size: " + $RESPONSE[0]
028  FGETLINES $Filename
029  PRINT "Number of lines: " + $RESPONSE[1]
030
031  PRINT "First 10 lines:"
032  PRINT ===================
033  CLEAR RESPONSE
```

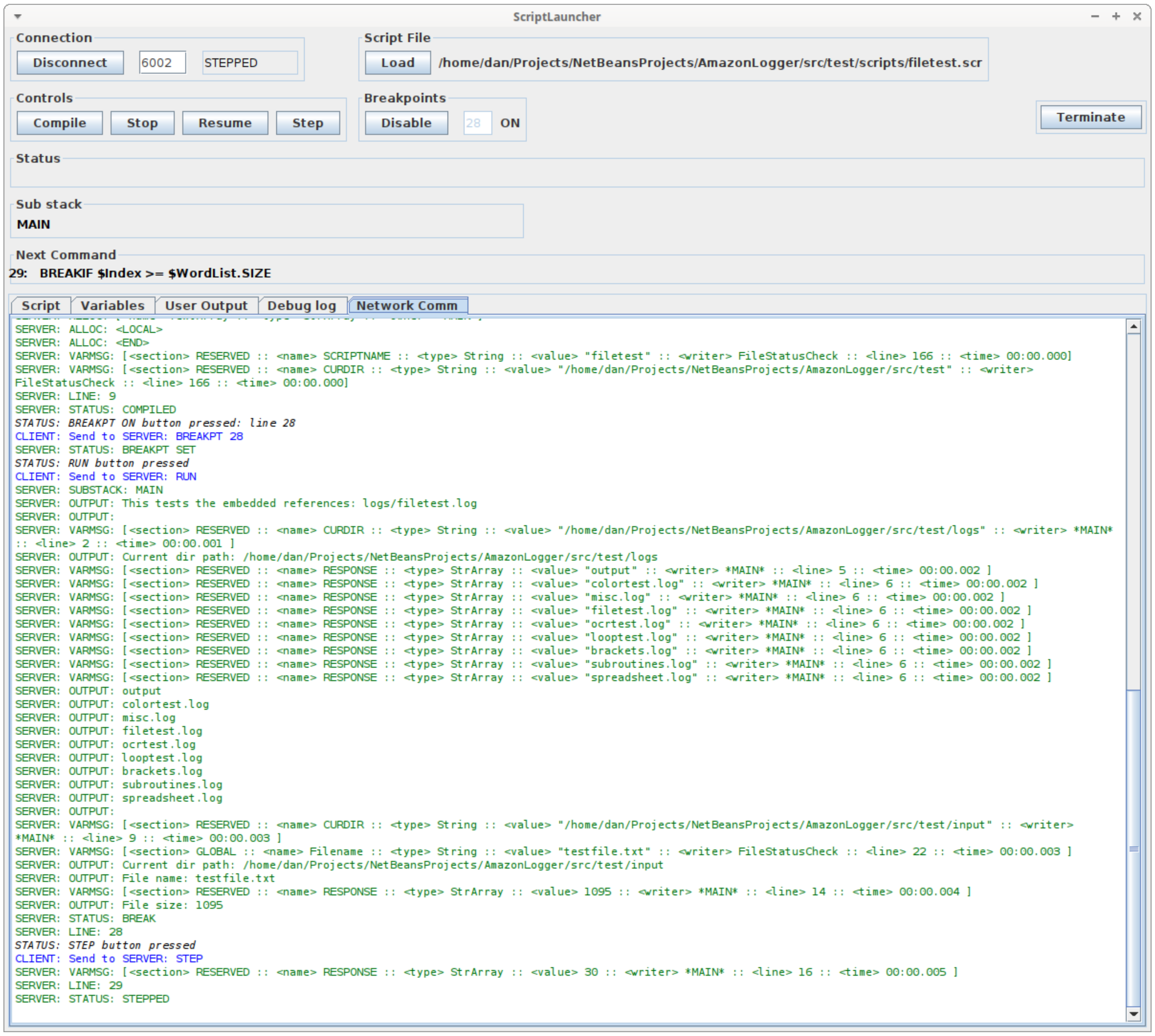
We can also look at the Debug log messages the server has sent in its corresponding tab.



```
00000549 00:00.076 [COMPIL]:    - IF level 1 ENDIF on line 123
00000550 00:00.076 [COMPIL]: PROGIX [124] (line 162): ENDIF
00000551 00:00.076 [COMPIL]:     line 124 ENDIF     for IF @ 116 (nest level 0)
00000552 00:00.076 [COMPIL]:    - IF level 0 ENDIF on line 124
00000553 00:00.076 [COMPIL]: PROGIX [125] (line 163): PRINT "Test success: " + $Success
00000554 00:00.076 [COMPIL]:      packed entry [0]: Discrete String: 'Test success: '
00000555 00:00.076 [COMPIL]:      packed entry [1]: Discrete String: '+'
00000556 00:00.076 [COMPIL]:      packed entry [2]: Reference String: '$Success'
00000557 00:00.076 [COMPIL]: PROGIX [126] (line 164): PRINT
00000558 00:00.076 [COMPIL]: PROGIX [127] (line 165): RETURN
00000559 00:00.076 [COMPIL]: PROGIX [128] (line 166): ENDSUB
00000560           [PROG  ]:     Running compiled file: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/scripts/filetest.scr (size 0 lines)
00000000           [VARS  ]:      LOCAL String Variable CheckType reset for subroutine: *MAIN*
00000001           [VARS  ]:      LOCAL String Variable Filename reset for subroutine: *MAIN*
00000002           [VARS  ]:      LOCAL StrArray Variable TextArray reset for subroutine: *MAIN*
00000003           [VARS  ]:      LOCAL String Variable Success reset for subroutine: FileStatusCheck
00000004           [PROG  ]: Resetting program index to begining
00000005           [PROG  ]: Script breakpoint set to line: 28
00000006 00:00.000 [PROG  ]: ===== BEGINING PROGRAM EXECUTION =====
00000007 00:00.000 [PROG  ]: PROGIX [0]: (line 9) PRINT
00000008 00:00.001 [PROG  ]:     Replaced arg value with expanded value: logs/filetest.log
00000009 00:00.001 [PROG  ]: PROGIX [1]: (line 10) PRINT
00000010 00:00.001 [PROG  ]: PROGIX [2]: (line 12) CD
00000011 00:00.001 [PROG  ]:     File path changed to: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs
00000012 00:00.001 [PROG  ]: PROGIX [3]: (line 13) PRINT
00000013 00:00.001 [PROG  ]: PROGIX [4]: (line 14) CLEAR
00000014 00:00.002 [VARS  ]:     - Deleted 0 entries in Array param: RESPONSE
00000015 00:00.002 [PROG  ]: PROGIX [5]: (line 15) DIRECTORY
00000016 00:00.002 [PROG  ]:     Path selection: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs
00000017 00:00.002 [PROG  ]: PROGIX [6]: (line 16) DIRECTORY
00000018 00:00.002 [PROG  ]:     Path selection: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs
00000019 00:00.002 [PROG  ]: PROGIX [7]: (line 17) PRINT
00000020 00:00.002 [PROG  ]: PROGIX [8]: (line 18) PRINT
00000021 00:00.002 [PROG  ]: PROGIX [9]: (line 20) CD
00000022 00:00.003 [PROG  ]: Backed up to directory above: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test
00000023 00:00.003 [PROG  ]:     File path changed to: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input
00000024 00:00.003 [PROG  ]: PROGIX [10]: (line 22) SET
00000025 00:00.003 [VARS  ]:      - Modified String param: Filename = testfile.txt
00000026 00:00.003 [PROG  ]: PROGIX [11]: (line 23) PRINT
00000027 00:00.003 [PROG  ]: PROGIX [12]: (line 24) PRINT
00000028 00:00.003 [VARS  ]:      GLOBAL String Variable Filename: testfile.txt
00000029 00:00.003 [PROG  ]: PROGIX [13]: (line 25) CLEAR
00000030 00:00.003 [VARS  ]:     - Deleted 9 entries in Array param: RESPONSE
00000031 00:00.003 [PROG  ]: PROGIX [14]: (line 26) FGETSIZE
00000032 00:00.003 [VARS  ]:      GLOBAL String Variable Filename: testfile.txt
00000033 00:00.004 [PROG  ]:     Path selection: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input/testfile.txt
00000034 00:00.004 [PROG  ]: PROGIX [15]: (line 27) PRINT
00000035 00:00.004 [VARS  ]:      RESPONSE[0] as type String: 1095
00000036 00:00.004 [PROG  ]: PROGIX [16]: (line 28) FGETLINES
00000037 00:00.004 [VARS  ]:      GLOBAL String Variable Filename: testfile.txt
00000038 00:00.004 [PROG  ]:     Path selection: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input/testfile.txt
```

And the Network Comm tab will show the communication messages between the server and the client.

Updated: 06/29/25



```
SERVER: ALLOC: <LOCAL>
SERVER: ALLOC: <END>
SERVER: VARMSG: [<section> RESERVED :: <name> SCRIPTNAME :: <type> String :: <value> "filetest" :: <writer> FileStatusCheck :: <line> 166 :: <time> 00:00.000]
SERVER: VARMSG: [<section> RESERVED :: <name> CURDIR :: <type> String :: <value> "/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test" :: <writer>
FileStatusCheck :: <line> 166 :: <time> 00:00.000]
SERVER: LINE: 9
SERVER: STATUS: COMPILED
STATUS: BREAKPT ON button pressed: line 28
CLIENT: Send to SERVER: BREAKPT 28
SERVER: STATUS: BREAKPT SET
STATUS: RUN button pressed
CLIENT: Send to SERVER: RUN
SERVER: SUBSTACK: MAIN
SERVER: OUTPUT: This tests the embedded references: logs/filetest.log
SERVER: OUTPUT:
SERVER: VARMSG: [<section> RESERVED :: <name> CURDIR :: <type> String :: <value> "/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs" :: <writer> *MAIN*
:: <line> 2 :: <time> 00:00.001 ]
SERVER: OUTPUT: Current dir path: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/logs
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "output" :: <writer> *MAIN* :: <line> 5 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "colortest.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "misc.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "filetest.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "ocrtest.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "looptest.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "brackets.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "subroutines.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> "spreadsheet.log" :: <writer> *MAIN* :: <line> 6 :: <time> 00:00.002 ]
SERVER: OUTPUT: output
SERVER: OUTPUT: colortest.log
SERVER: OUTPUT: misc.log
SERVER: OUTPUT: filetest.log
SERVER: OUTPUT: ocrtest.log
SERVER: OUTPUT: looptest.log
SERVER: OUTPUT: brackets.log
SERVER: OUTPUT: subroutines.log
SERVER: OUTPUT: spreadsheet.log
SERVER: OUTPUT:
SERVER: VARMSG: [<section> RESERVED :: <name> CURDIR :: <type> String :: <value> "/home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input" :: <writer>
*MAIN* :: <line> 9 :: <time> 00:00.003 ]
SERVER: VARMSG: [<section> GLOBAL :: <name> Filename :: <type> String :: <value> "testfile.txt" :: <writer> FileStatusCheck :: <line> 22 :: <time> 00:00.003 ]
SERVER: OUTPUT: Current dir path: /home/dan/Projects/NetBeansProjects/AmazonLogger/src/test/input
SERVER: OUTPUT: File name: testfile.txt
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> 1095 :: <writer> *MAIN* :: <line> 14 :: <time> 00:00.004 ]
SERVER: OUTPUT: File size: 1095
SERVER: STATUS: BREAK
SERVER: LINE: 28
STATUS: STEP button pressed
CLIENT: Send to SERVER: STEP
SERVER: VARMSG: [<section> RESERVED :: <name> RESPONSE :: <type> StrArray :: <value> 30 :: <writer> *MAIN* :: <line> 16 :: <time> 00:00.005 ]
SERVER: LINE: 29
SERVER: STATUS: STEPPED
```