//

Debugging デバッグ

maXbox Starter 146 – Get a Debugger.

"Se contentus est sapiens. [1] – Seneca.

Source: **1413_services5jcl_1.pas**
1190_Continued_fraction64_python3.12.4debug30.txt

https://sourceforge.net/projects/maxbox5/files/examples/1413_services5jcl.pas/download

https://sourceforge.net/projects/maxbox5/files/EKON29/1190_Continued_fraction64_python3.12.4debug30.txt/download

1406_U_Go3_1form2.pas
1413_services5jcl.pas

Every programmer spends a lot of time fixing bugs. Whether it's a small typo or a tricky logic error, knowing how to debug efficiently is an important skill.
But debugging isn't just about fixing mistakes—it's also about understanding what went wrong and learning how to avoid similar problems in the future.
In maXbox or Delphi we had to connect one event from **TPSScript** to the cdebugAfterExecute Event:

```
object PSScript: TPSScript
    CompilerOptions = [icAllowUnit]
    OnLine = PSScriptLine
    OnCompile = PSScriptCompile
    OnExecute = PSScriptExecute
    OnAfterExecute = cedebugAfterExecute
    OnCompImport = IFPS3ClassesPlugin1CompImport
    OnExecImport = IFPS3ClassesPlugin1ExecImport
    Plugins = <
      item
        Plugin = PS3DllPlugin
      end>
    UsePreProcessor = True
    OnNeedFile = PSScriptNeedFile
    Left = 392
    Top = 136
  end
```

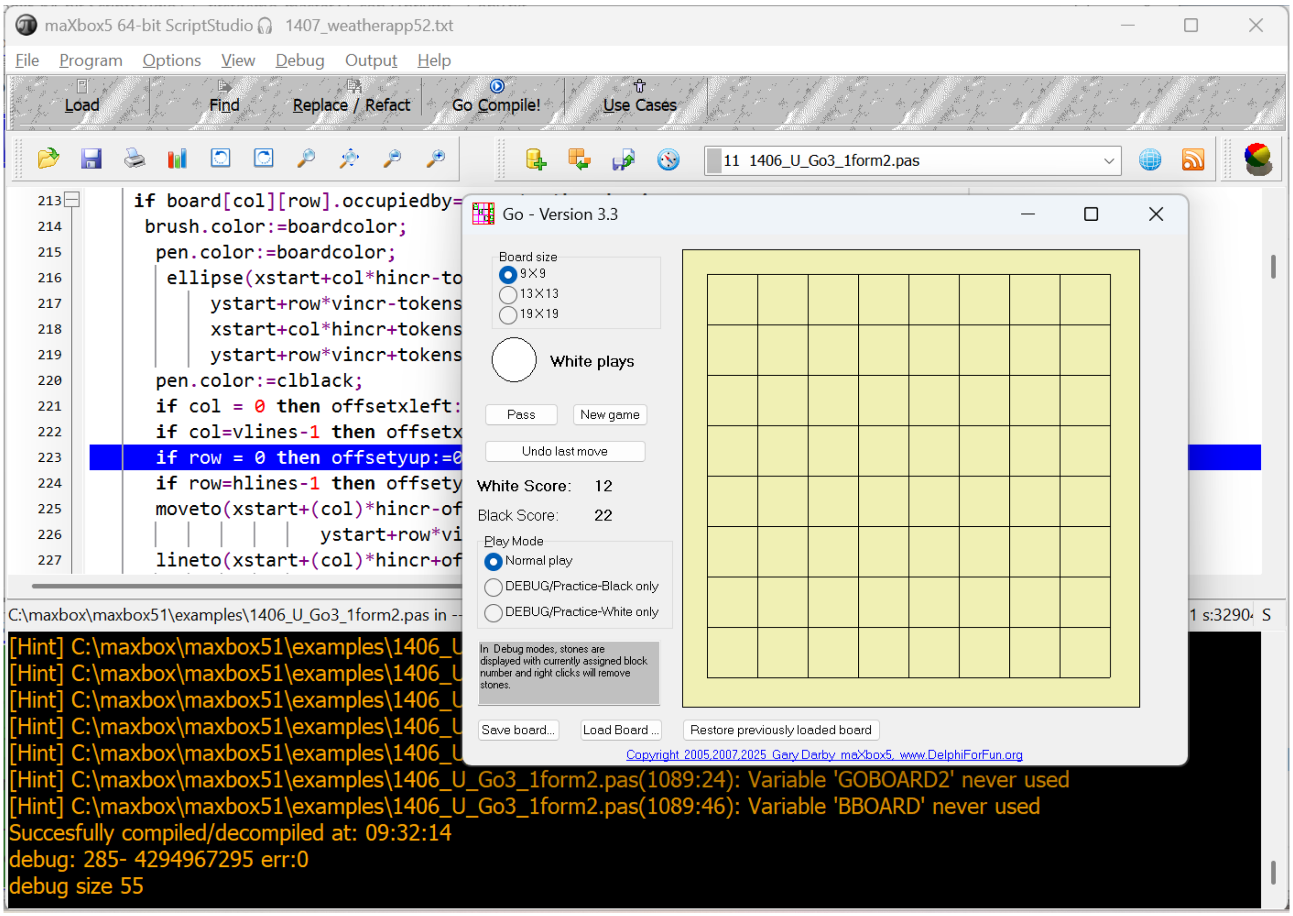
The debug engine (DE) typically sends one of the following events as the last startup event:

- The entry point event, if attaching to a newly launched program

1 (ep. 9, 13) - Der Weise ist zufrieden mit sich selbst. -

- The load complete event, if attaching to a program that is already running.

If the debug session with **F8** is to ignore a particular stopping event, the debug session calls the program's Continue method. If the program was stepping into, over, or out of a function when it encountered the stopping condition, then it continues the step.


maXbox5 64-bit ScriptStudio 1407_weatherapp52.txt
File Program Options View Debug Output Help
Load
Find
Replace / Refact
Go Compile!
Use Cases
11 1406_U_Go3_1form2.pas
Go - Version 3.3
Board size
9×9
13×13
19×19
White plays
Pass
New game
Undo last move
Play Mode
Normal play
DEBUG/Practice-Black only
DEBUG/Practice-White only
Save board...
Load Board ...
Restore previously loaded board
Copyright 2005,2007,2025 Gary Darby maXbox5 www.DelphiForFun.org

```
if board[col][row].occupiedby=
 brush.color:=boardcolor;
  pen.color:=boardcolor;
   ellipse(xstart+col*hincr-to
       ystart+row*vincr-tokens
       xstart+col*hincr+tokens
       ystart+row*vincr+tokens
  pen.color:=clblack;
  if col = 0 then offsetxleft:
  if col=vlines-1 then offsetx
  if row = 0 then offsetyup:=0
  if row=hlines-1 then offsety
  moveto(xstart+(col)*hincr-of
                ystart+row*vi
  lineto(xstart+(col)*hincr+of
```

White Score: 12
Black Score: 22
[✓ Normal play]
In Debug modes, stones are displayed with currently assigned block number and right clicks will remove stones.

```
C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas in --
[Hint] C:\maxbox\maxbox51\examples\1406_U
[Hint] C:\maxbox\maxbox51\examples\1406_U
[Hint] C:\maxbox\maxbox51\examples\1406_U
[Hint] C:\maxbox\maxbox51\examples\1406_U
[Hint] C:\maxbox\maxbox51\examples\1406_U
[Hint] C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas(1089:24): Variable 'GOBOARD2' never used
[Hint] C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas(1089:46): Variable 'BBOARD' never used
Succesfully compiled/decompiled at: 09:32:14
debug: 285- 4294967295 err:0
debug size 55
```

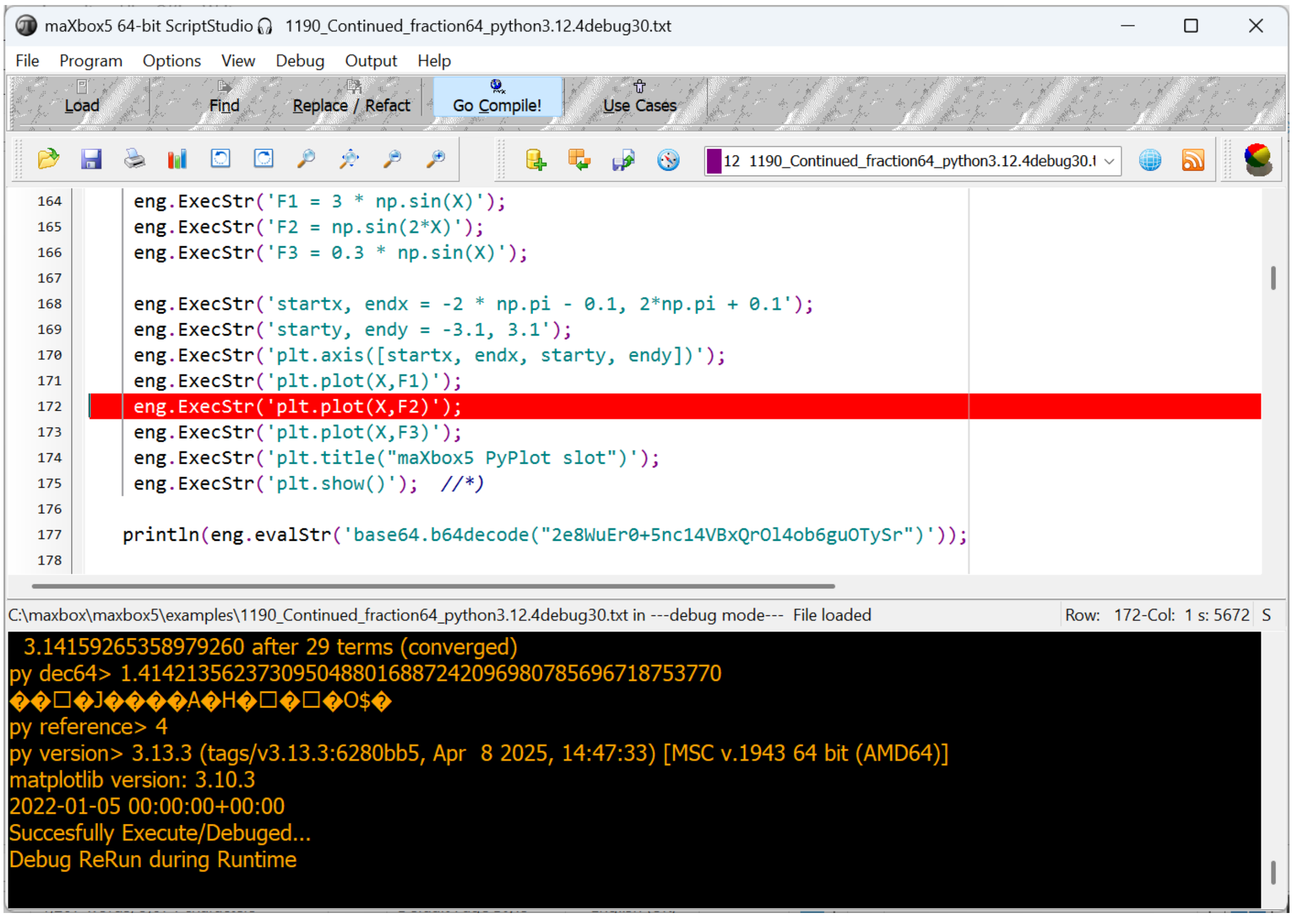
1406_livedebugger1_Screenshot2025-06-19_093305.png

Debuging and Decompile can be disturbed either by events or method handlers are in use (e.g. onClose, onDrawcell, onIdle..) or a property like form.style =[fsBold]. So uncomment this line to debug the code.

If the user elects to step into **F7**, over **F8**, or out of a function **Ctrl+ F9**, the IDE prompts the debug session to call the program's Step method. The IDE then passes the unit of step (instruction, statement, or line) and the type of step (whether to step into, over, or out of the function). When the step is complete, the DE sends a step complete event to the debug session, which is a stopping event.
If with F8 as step over goes into a loop it works like a live debugger in the script and you can feel where to code is running!

The program then runs until it encounters a stopping condition, such as hitting a breakpoint set with **F5**. In which case, the DE sends a breakpoint event to the debug session. The breakpoint event is a stopping event, and the DE again waits for a user response.

When you set a breakpoint with F5 (you can set more than one) then you start the debug session with Debug Run till the breakpoint it waits, then you step further with Debug Run or **Ctrl+ F9** like continue. If the debug session is to ignore a particular stopping event, the debug session calls the program's Continue method. If the program was stepping into, over, or out of a function when it encountered the stopping condition, then it continues the step.


maXbox5 64-bit ScriptStudio 1190_Continued_fraction64_python3.12.4debug30.txt
File Program Options View Debug Output Help
Load
Find
Replace / Refact
Go Compile!
Use Cases
12 1190_Continued_fraction64_python3.12.4debug30.t

```
    eng.ExecStr('F1 = 3 * np.sin(X)');
    eng.ExecStr('F2 = np.sin(2*X)');
    eng.ExecStr('F3 = 0.3 * np.sin(X)');

    eng.ExecStr('startx, endx = -2 * np.pi - 0.1, 2*np.pi + 0.1');
    eng.ExecStr('starty, endy = -3.1, 3.1');
    eng.ExecStr('plt.axis([startx, endx, starty, endy])');
    eng.ExecStr('plt.plot(X,F1)');
    eng.ExecStr('plt.plot(X,F2)');
    eng.ExecStr('plt.plot(X,F3)');
    eng.ExecStr('plt.title("maXbox5 PyPlot slot")');
    eng.ExecStr('plt.show()');  //*)

   println(eng.evalStr('base64.b64decode("2e8WuEr0+5nc14VBxQrOl4ob6guOTySr")'));
```

C:\maxbox\maxbox5\examples\1190_Continued_fraction64_python3.12.4debug30.txt in ---debug mode--- File loaded   Row: 172-Col: 1 s: 5672 S

```
 3.14159265358979260 after 29 terms (converged)
py dec64> 1.4142135623730950488016887242096980785696718753770
��□�J����A�H�□�□�O$�
py reference> 4
py version> 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bit (AMD64)]
matplotlib version: 3.10.3
2022-01-05 00:00:00+00:00
Succesfully Execute/Debuged...
Debug ReRun during Runtime
```

1413_services2_mx5_screenshot.png //1406_1_GoScreenshot2025-06-162815.png

For the advanced setting of the debugger we take a look to the component **TPSScriptDebugger** with the corresponding events.

```
object cedebug: TPSScriptDebugger
    CompilerOptions = [icAllowUnit]
    OnCompile = PSScriptCompile
    OnExecute = cedebugExecute
    OnAfterExecute = cedebugAfterExecute
    OnCompImport = IFPS3ClassesPlugin1CompImport
    OnExecImport = IFPS3ClassesPlugin1ExecImport
    Plugins = <
      item
        Plugin = PS3DllPlugin
      end
      item
      end>
    UsePreProcessor = True
    OnNeedFile = PSScriptNeedFile
    OnIdle = cedebugIdle
    OnLineInfo = cedebugLineInfo
    OnBreakpoint = cedebugBreakpoint
    Left = 344
    Top = 136
  end
```

Also set the breakpoint in a single function works:

```
function StartServiceByName(const AServer,AServiceName: String):Boolean;
var
  ServiceHandle,
  SCMHandle: SC_HANDLE;
  P: PChar;
begin
  P:= nil;
  Result:= False;
  SCMHandle:= OpenSCManager(PChar(AServer), nil, SC_MANAGER_ALL_ACCESS);
  if SCMHandle <> 0 then
  try
    ServiceHandle:= OpenService(SCMHandle, Pchar(AServiceName),
                                                 SERVICE_ALL_ACCESS);
    if ServiceHandle <> 0 then
      Result:= StartService(ServiceHandle, 0, P);

    CloseServiceHandle(ServiceHandle);
  finally
    CloseServiceHandle(SCMHandle);
  end;
end;
```

We call that from the main:

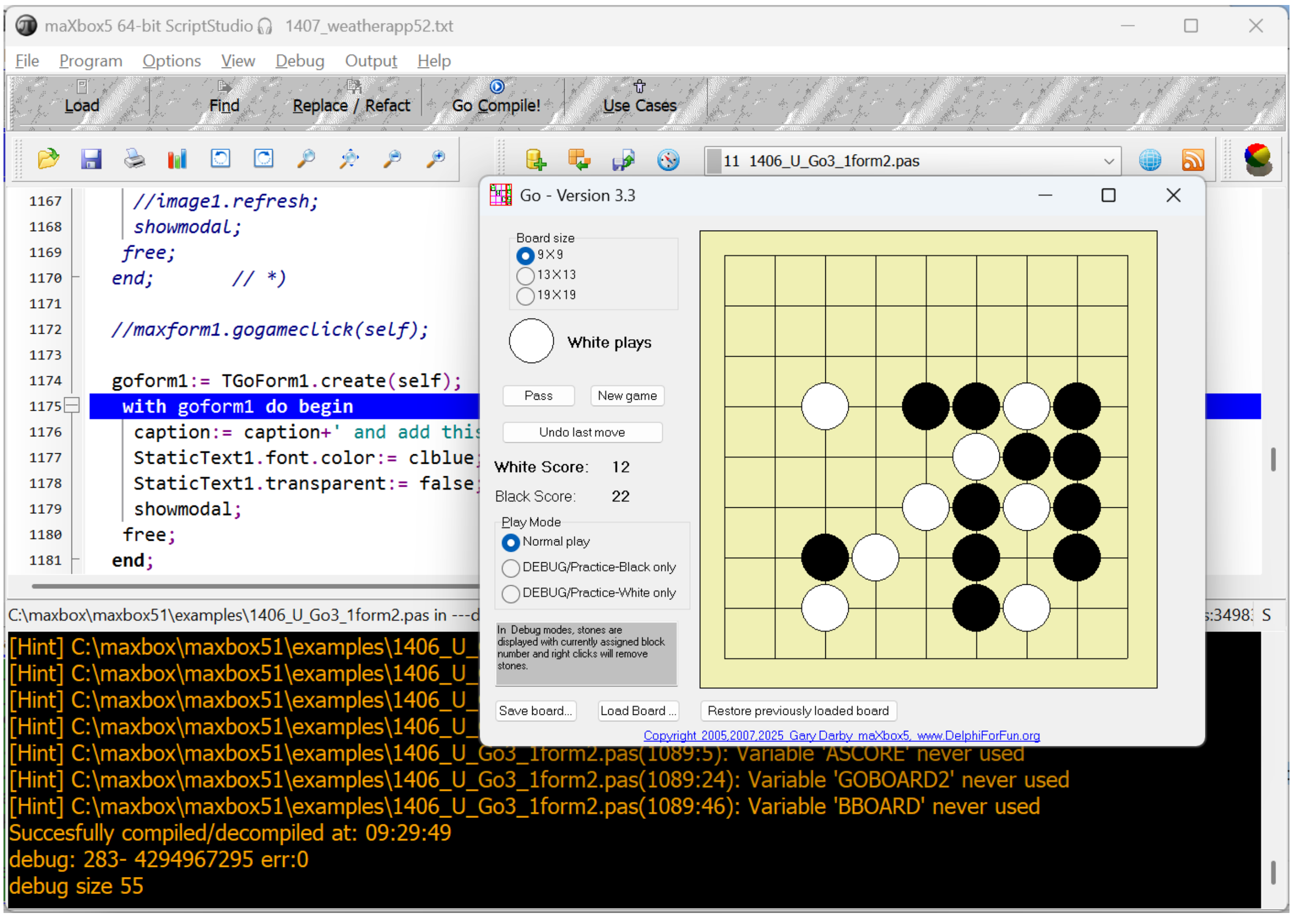
```
if StartServiceByName('DESKTOP-BTLKHKF','ALG') then
writ('ALG started...');
writ('stat of ALG '+
     itoa(ord(GetServiceStatusByName('DESKTOP-BTLKHKF','ALG'))));
  sleep(500)
  writ('stat of ALG '+
              itoa(ord(GetServiceStatusByName('DESKTOP-BTLKHKF','ALG'))));
  //toogle 2____________
  //StopServiceByName('DESKTOP-BTLKHKF','ALG');
  sleep(500)
```

Between the two sleeps you can see two different states, namely 2 and 4:

```
//GetServiceStatusWaitingIfPending(ServiceHandle: SC_HANDLE): DWORD;
 {(ssUnknown,         // Just fill the value 0
    ssStopped,          // SERVICE_STOPPED
    ssStartPending,     // SERVICE_START_PENDING
    ssStopPending,      // SERVICE_STOP_PENDING
    ssRunning,          // SERVICE_RUNNING
    ssContinuePending, // SERVICE_CONTINUE_PENDING
    ssPausePending,     // SERVICE_PAUSE_PENDING
    ssPaused);          // SERVICE_PAUSED }
```

So it changes from *ssStartPending* ***2*** to *ssRunning* ***4,*** on methods see first screenshot above.

Besides using the previous methods, if you need more control in a debugger way you can use WMI. With Win32_Service class you have access to all information of the services installed on the machine and you can has access to methods: Start, Stop, Pause, Resume, Interrogate, Create, Delete, Change, ChangeStartMode…



```
    //image1.refresh;
    showmodal;
   free;
  end;       // *)

  //maxform1.gogameclick(self);

  goform1:= TGoForm1.create(self);
   with goform1 do begin
    caption:= caption+' and add this
    StaticText1.font.color:= clblue;
    StaticText1.transparent:= false;
    showmodal;
   free;
  end;
```

C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas in ---d

```
[Hint] C:\maxbox\maxbox51\examples\1406_U_
[Hint] C:\maxbox\maxbox51\examples\1406_U_
[Hint] C:\maxbox\maxbox51\examples\1406_U_
[Hint] C:\maxbox\maxbox51\examples\1406_U_
[Hint] C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas(1089:5): Variable 'ASCORE' never used
[Hint] C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas(1089:24): Variable 'GOBOARD2' never used
[Hint] C:\maxbox\maxbox51\examples\1406_U_Go3_1form2.pas(1089:46): Variable 'BBOARD' never used
Succesfully compiled/decompiled at: 09:29:49
debug: 283- 4294967295 err:0
debug size 55
```

Board size: [✓ 9×9] 13×13 19×19

White plays

White Score: 12

Black Score: 22

Play Mode: [✓ Normal play] DEBUG/Practice-Black only DEBUG/Practice-White only

In Debug modes, stones are displayed with currently assigned block number and right clicks will remove stones.

Copyright 2005,2007,2025 Gary Darby maXbox5, www.DelphiForFun.org

1406_livedebugger2_Screenshot2025-06-19_093052.png
1413_services_mx5_screenshot.png

How TPSScriptDebugger Works in Pascal Script

TPSScriptDebugger is a component in RemObjects Pascal Script designed to provide debugging capabilities for scripts executed within your Delphi, maXbox or C++Builder applications. While the search results do not provide a direct, detailed breakdown of TPSScriptDebugger, the general workflow of debugging in Pascal Script and the integration of debugging support can be inferred from the available information and standard usage patterns of the library.

General Workflow of Pascal Script Execution

- **Script Compilation:** Scripts are first compiled using the TPSPascalCompiler class. The script is parsed and converted into bytecode, and any compiler messages (errors or warnings) are collected during this stage 1.

- **Script Execution:** The compiled bytecode is then executed by the TPSExec class. If runtime errors occur, they are captured and reported, typically using methods like PSErrorToString to convert error codes to readable messages12.

Debugging Support

- **Debugger Integration:** TPSScriptDebugger works by hooking into the script execution process. It allows you to:
  - Set breakpoints in your script code.
  - Step through script execution line by line.
  - Inspect and modify variable values at runtime.
  - Monitor the call stack and execution flow.
- **Usage Pattern:**
  - You typically associate a TPSScriptDebugger instance with your TPSScript or TPSExec instance.
  - The debugger listens for execution events (such as line changes or exceptions) and provides callbacks or events that your host application can use to update the UI (e.g., highlighting the current line, showing variable values).
  - When a breakpoint is hit, execution is paused, and control is returned to the host application, allowing inspection and step operations.
- **Implementation Details:**
  - TPSScriptDebugger interacts closely with the script engine, receiving notifications about instruction pointer changes, exceptions, and variable state.
  - The component exposes methods and events for common debugging actions: continue, step over, step into, step out, and stop.
  - It maintains a mapping between script source lines and bytecode instructions, enabling accurate breakpoints and stepping.

Example Usage

While the search results do not include a full code example for TPSScriptDebugger, the general approach is as follows (pseudocode):

```
text
var Script: TPSScript;
  Debugger: TPSScriptDebugger;
begin
  Script := TPSScript.Create(nil);
  Debugger := TPSScriptDebugger.Create(nil);
  Script.Debugger := Debugger;
  // Load and compile script
  // Set breakpoints via Debugger.Breakpoints.Add(lineNumber)
  // Start execution; Debugger events will handle breakpoints and
                                                            stepping
end;
```

Key Features

- **Breakpoints:** Set and remove breakpoints at specific source lines.
- **Stepping:** Step into, over, or out of script code.
- **Variable Inspection:** View and modify script variables during paused execution.
- **Call Stack:** Inspect the current call stack to understand the execution context.

Summary Table: TPSScriptDebugger Capabilities

Feature	Description
Breakpoints	Pause execution at specific script lines
Stepping	Step into, over, or out of code
Variable Inspection	View/modify script variables during debugging
Call Stack	Inspect current function/procedure call stack
Exception Handling	Catch and report script exceptions

References

- For a general overview of Pascal Script compilation and exe: 12
- For integration and debugging capabilities: 12

If you need more detailed, code-level documentation for TPSScriptDebugger, consulting the official RemObjects Pascal Script documentation or the source code itself is recommended, as community resources often provide only high-level overviews12.

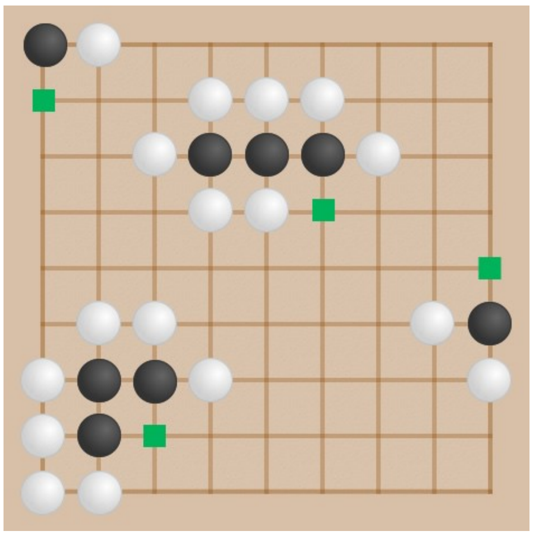
1. https://lawrencebarsanti.wordpress.com/2009/11/28/introduction-to-pascal-script/
2. https://wiki.freepascal.org/Pascal_Script_Examples
3. https://talk.remobjects.com/t/pascal-script-set-var-question/10074
4. https://stackoverflow.com/questions/17101532/loading-pre-compiled-script-in-remobjects-pascal-script-delphi
5. https://doc.tmssoftware.com/biz/scripter/scripter-user-guide.pdf
6. https://doc.tmssoftware.com/biz/scripter/guide/scripter.html
7. https://www.pascalgamedevelopment.com/showthread.php?32620-Scripting-with-remote-debugger
8. https://www.youtube.com/watch?v=9dnGDI8qHsY

A last and interesting call concerns the shell with a kind of debug. If you are familiar with the console just do an external console call from the script:

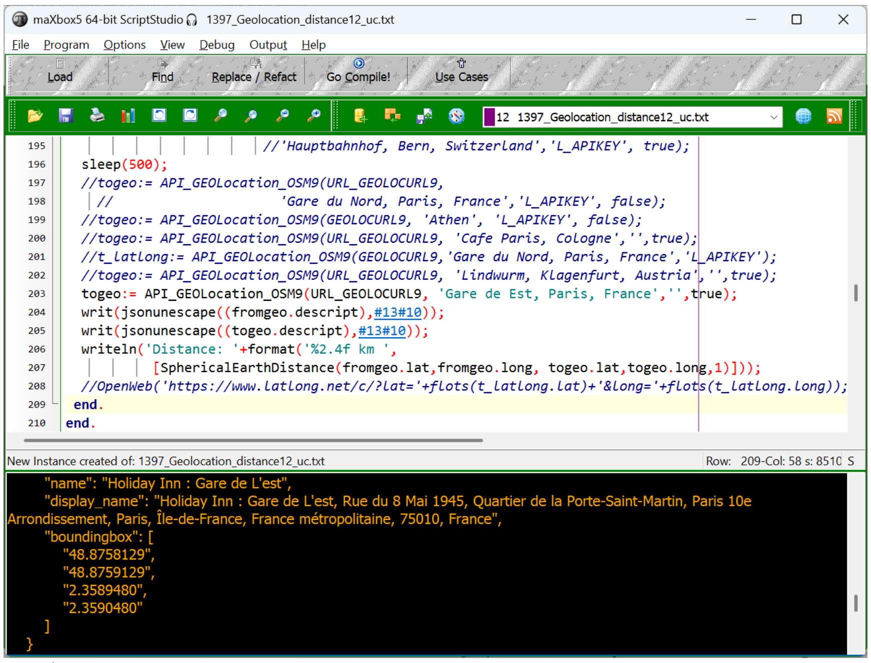
```
ShellExecute(0, nil, 'cmd.exe', 'sc config "the service name"
                                       start=disabled', nil, SW_HIDE);
ShellExecute(0, nil, 'cmd.exe', 'sc config "the service name"
                                       start=auto', nil, SW_HIDE);

RunDosInMemo('cmd.exe /C sc config "ALG" start= disabled',Memo2);
>>> [SC] ChangeServiceConfig SUCCESS
RunDosInMemo('cmd.exe /C sc query "ALG"',Memo2);
```

The **sc config** command is a powerful tool in Windows that allows you to modify the configuration of a service. This command can change various parameters of a service, such as its start type, error control, binary path, dependencies, and more. It is particularly useful for system administrators who need to manage services on local or remote machines.

Another **Appendix** Page



In September, Brian Kleiner represented Switzerland in the 17th Korea Prime Minister Cup (KPMC), in Gwangju. There were players from 53 countries present, with 6 rounds over three days of play. The organi-sation from the hosts was excellent from start to finish, the late summer weather was ideal, and a good spirit reigned throughout.


maXbox5 64-bit ScriptStudio 1397_Geolocation_distance12_uc.txt
File Program Options View Debug Output Help
Load Find Replace / Refact Go Compile! Use Cases
12 1397_Geolocation_distance12_uc.txt

```
                //'Hauptbahnhof, Bern, Switzerland','L_APIKEY', true);
  sleep(500);
  //togeo:= API_GEOLocation_OSM9(URL_GEOLOCURL9,
   //                     'Gare du Nord, Paris, France','L_APIKEY', false);
  //togeo:= API_GEOLocation_OSM9(GEOLOCURL9, 'Athen', 'L_APIKEY', false);
  //togeo:= API_GEOLocation_OSM9(URL_GEOLOCURL9, 'Cafe Paris, Cologne','',true);
  //t_latlong:= API_GEOLocation_OSM9(GEOLOCURL9,'Gare du Nord, Paris, France','L_APIKEY');
  //togeo:= API_GEOLocation_OSM9(URL_GEOLOCURL9, 'Lindwurm, Klagenfurt, Austria','',true);
  togeo:= API_GEOLocation_OSM9(URL_GEOLOCURL9, 'Gare de Est, Paris, France','',true);
  writ(jsonunescape((fromgeo.descript),#13#10));
  writ(jsonunescape((togeo.descript),#13#10));
  writeln('Distance: '+format('%2.4f km ',
          [SphericalEarthDistance(fromgeo.lat,fromgeo.long, togeo.lat,togeo.long,1)]));
  //OpenWeb('https://www.latlong.net/c/?lat='+flots(t_latlong.lat)+'&long='+flots(t_latlong.long));
 end.
end.
```

New Instance created of: 1397_Geolocation_distance12_uc.txt   Row: 209-Col: 58 s: 8510 S

```
    "name": "Holiday Inn : Gare de L'est",
    "display_name": "Holiday Inn : Gare de L'est, Rue du 8 Mai 1945, Quartier de la Porte-Saint-Martin, Paris 10e
Arrondissement, Paris, Île-de-France, France métropolitaine, 75010, France",
    "boundingbox": [
      "48.8758129",
      "48.8759129",
      "2.3589480",
      "2.3590480"
    ]
  }
```

MaxMatrix Time/Space:

The multiplication of past x future is a vector with the function:= known = f(changeable) [y=f(x)] as distance over time, so **distance** is a function of time: **d=f(t)**.

Conclusion

While the search results do not mention TPSScriptDebugger specifically, standard script debuggers—including those for Pascal Script—share core features that facilitate tracing the execution flow of scripts. These features, as seen in various script debuggers, typically include:

Breakpoints: Allow you to pause script execution at specific lines, so you can examine the state of the script and its variables at critical points. Step Over, Step Into, Step Out: These commands let you control execution line by line:

Step Over: Executes the current line and moves to the next, skipping over function calls.
Step Into: Dives into the details of a function call, allowing you to trace execution inside called functions or procedures.
Step Out: Runs the rest of the current function and pauses execution when control returns to the caller.

The **sc config** command is a powerful tool in Windows that allows you to modify the configuration of a service.

Script:

https://sourceforge.net/projects/maxbox5/files/examples/1413_services5jcl_1.pas/download

References:

Advanced Debugging API

How to use the command 'sc' (with examples)

🐞 圍棋.

Doc and Tool: maXbox5 - Manage Files at SourceForge.net

Max Kleiner 19/06/2025