

Brian's Diary and Wiki 2015-04-24

Last modified on 27/04/2015 10:07 by Brian THOMAS.
Tags:

2015-04-24

Cases 2216, 2230 & 2231

Porter Dodson - all these defects were intermittent - they were reported but when I remoted in the following day the defect wasn't there.

Case 2153

More Delphi debugging.

At DLL_PROCESS_DETACH time KappSrvr suffers an access violation at line 7 in this function from **JclUnitVersioning.pas**:

```
1: procedure UnregisterUnitVersion(Instance: THandle);
2: var
3:   UnitVersioning: TUnitVersioning;
4: begin
5:   UnitVersioning := GetUnitVersioning;
6:   if Assigned(UnitVersioning) then
7:     UnitVersioning.UnregisterModule(Instance);
8: end;
```

The exception looks like this:

```
-----
Debugger Exception Notification
-----
Project winword.exe raised exception class $C0000005 with message 'access violation at 0x1aa07b69: read of address 0x1069a420'.
-----
Break   Continue   Help
-----
```

And the call stack looks like this:

```
JclUnitVersioning.UnregisterUnitVersion(444137472)
JclFileUtils.Finalization
System.FinalizeUnits
System._Halt0
SysInit._InitLib
:1a799e2d @Halt0 + $B1
:77219ea6 ; C:\Windows\System32\ntdll.dll
:77219e22 ntdll.RtlInitializeCriticalSection + 0xa2
:7723da28 ; C:\Windows\System32\ntdll.dll
:7723dad1 ntdll.RtlExitUserProcess + 0x81
:74d89863 kernel32.ExitProcess + 0x13
:5e1e6148 ; C:\Program Files\Microsoft Office 15\root\office15\wwlib.dll
:5e16da11 ; C:\Program Files\Microsoft Office 15\root\office15\wwlib.dll
:5e16de83 ; C:\Program Files\Microsoft Office 15\root\office15\wwlib.dll
:5ddc615c ; C:\Program Files\Microsoft Office 15\root\office15\wwlib.dll
:012115c4 ; C:\Program Files\Microsoft Office 15\root\office15\winword.exe
:01211558 ; C:\Program Files\Microsoft Office 15\root\office15\winword.exe
:74d77c04 kernel32.BaseThreadInitThunk + 0x24
:7722b90f ntdll.RtlInitializeExceptionChain + 0x8f
:7722b8da ntdll.RtlInitializeExceptionChain + 0x5a
```

System.pas:FinalizeUnits has a **try** block that calls each unit's **Finalize** method and its **catch** block also calls **FinalizeUnits**. This explains the recursive calls I saw previously, where each time the breakpoint hit there was one extra **FinalizeUnits** item in the call stack. Starting at the tail end of the following table, each time **FinalizeUnits** finalizes a unit it decrements an index, so if the **try** block does throw an exception, the subsequent call to **FinalizeUnits** from inside the **catch** block will restart at the unit preceding the one that just failed.

The table that gets walked to run the Initialize and Finalize sections for each unit is declared in **System.pas**:

```
PackageUnitEntry = packed record
  Init, FInit : Pointer;
end;

{ Compiler generated table to be processed sequentially to init & finit all package units }
{ Init: 0..Max-1; Final: Last Initialized..0 }
UnitEntryTable = array [0..9999999] of PackageUnitEntry;
PUnitEntryTable = ^UnitEntryTable;
```

Memory 1			
1AC20980	1AC2CC90	1AB8F574	.iÄ.tö.
1AC20988	1AC2CCB8	1AB8F5F4	.iÄ.öö.
1AC20990	00000000	00000000
1AC20998	1AC2CCDC	1AB8F670	üiÄ.pö.
1AC209A0	1AC2CCFC	1AB8F6E4	üiÄ.öö.
1AC209A8	1AC2CD1C	1AB90490	.fÄ...¹.
1AC209B0	1AC2CD3C	1AB906B0	<fÄ.º.¹.
1AC209B8	1AC2CD5C	1AB90730	.fÄ.0.¹.
1AC209C0	1AC2CD7C	1AB9142C	fÄ.,.¹.
1AC209C8	1AC2CDA8	1AB92240	"fÄ.@".¹.
1AC209D0	1AC2CDF0	1AB92420	öfÄ.\$¹.
1AC209D8	1AC2CE10	1AB92E00	.fÄ...¹.
1AC209E0	1AC2CE38	1AB93D6C	8fÄ.l=¹.
1AC209E8	1AC2CE60	1AB94A00	.fÄ..J¹.
1AC209F0	1AC2CE80	1AB94A78	€fÄ.xJ¹.
1AC209F8	1AC2CEA0	1AB95358	.fÄ.XS¹.
1AC20A00	1AC2CEC4	1AB976D0	ÄfÄ.öV¹.
1AC20A08	00000000	00000000
1AC20A10	00000000	00000000
1AC20A18	00000000	00000000
1AC20A20	00000000	00000000
1AC20A28	1AC2CEE4	1AB983A8	äfÄ."f¹.
1AC20A30	1AC2CEF8	1AB989F8	öfÄ.ö§¹. [272]
1AC20A38	00000000	00000000
1AC20A40	00000000	00000000
1AC20A48	00000000	00000000
1AC20A50	00000000	00000000
1AC20A58	1AC2CF28	1AB9CF08	(fÄ..I¹.
1AC20A60	1AC2CF54	1ABA1624	TfÄ.\$..e. [278]
1AC20A68	00000000	00000000
1AC20A70	1AC2CF78	1ABD8AE8	xifÄ.è§%.
1AC20A78	00000000	00000000
1AC20A80	00000000	00000000
1AC20A88	1AC2D004	1ABED258	.öÄ.X0%.
1AC20A90	00000000	00000000
1AC20A98	00000000	00000000
1AC20AA0	00000000	00000000
1AC20AA8	00000000	00000000
1AC20AB0	00000000	00000000
1AC20AB8	00000000	00000000
1AC20AC0	00000000	00000000
1AC20AC8	00000000	00000000
1AC20AD0	00000000	00000000
1AC20AD8	1AC2D020	1AC1A044	öÄ.D Ä. [293]
1AC20AE0	00000000	00000000
1AC20AE8	00000000	00000000
1AC20AF0	1AC2D028	1AC1FB90	(öÄ..üÄ.
1AC20AF8	00000000	00000000
1AC20B00	00000000	00000000
1AC20B08	00000000	1AC20194	...".Ä.
1AC20B10	00000001	1AC2EC34	...4iÄ.

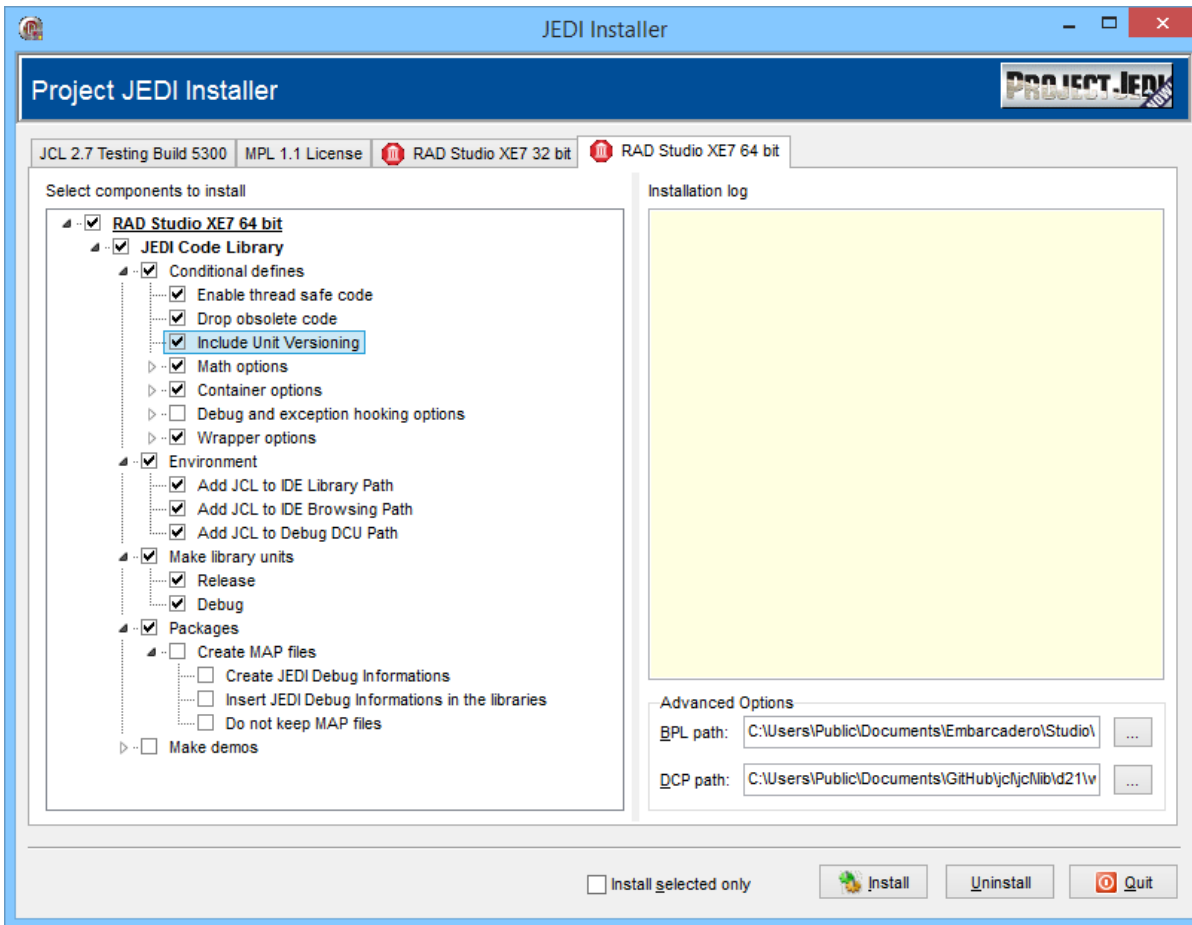
This is the dump of the compiler-generated table. The KappSrvr entry is the pair of DWORDs at [272]. The address of the Finit function which calls on to the KappSrvr Finalization section is 1AB989F8.

```

1:// from Utils.pas
2://
3: initialization
4:     InitializeCriticalSection(CritSect);
5:     Profiler := TCodeSiteLogger.Create(nil);
6:     Profiler.Enabled := False;
7:
8: finalization
9:     DeleteCriticalSection(CritSect);
10:    Profiler.AddResetSeparator;
11:    Profiler.Free;
12:    Profiler := nil;
13:
14: end.
```

Starting at index 299, for the first few indices this runs fine. But entry [266] is the finalization block for JclUnitVersioning.pas itself.

I've taken a further VM snapshot, and I'm going to rebuild the JCL libraries with **UnitVersioning** turned off. The snap shows the default settings (which I used when I first installed the JEDI stuff).



I tried this twice - first time around I unchecked both [x] Include Unit Versioning and [x] Environment because I didn't want it to break the environment I have already. But Delphi squawked on start up, so on the second attempt I reinstated Environment and did the build again. But the errors are still there - I'll have to fix that next week.

The rebuilt KappSrvr allowed Winword to exit cleanly without crashing at exit. Yay!