Twitter Thread by Anson Horton



Anson Horton
@AnsonHorton

1/ @werat asked about whether the debugger was using the C# compiler or language service in VS 2002. It was not. The debugger has a component called an 'expression evaluator' that is provided per language and is responsible for parsing and evaluating expressions when stopped at a

2/ breakpoint. For example, if you type into the immediate window, hover over a variable, type into the watch window, etc. the expression evaluator is involved. The debugger and the language service are actually deeply integrated in a number of scenarios in VS, which may

3/ initially seem surprising. I may talk about more of these scenarios in the future, but to give a flavor, when you set a breakpoint at design time the language service is involved, when you are using Edit and Continue the LS is involved, the range of what is being evaluated

4/ when you get a debugger data tip, completion lists when typing in the data windows or immediate window, etc. Prior to VS 2002, there were distinct debug environments tailored to each target customer (VB/VBA, C++, VJ, VBScript, TSQL, FoxPro, Fortran, etc.). The goals of the

5/ VS 2002 debugger included unifying the debugger backend components and providing a single UI stack that would enable new scenarios (for example, a callstack across languages). Expression evaluators are the architecture that separate the language specific pieces from the

6/ agnostic parts. However, as the C# team was comparatively small, in VS 2002, the debugger team implemented the expression evaluator for C#. The debugger team decided to implement a single expression evaluator that would cover 'managed C++' and C#, which made sense originally

7/ likely as it wasn't clear how different C# would be. Unfortunately, the ownership and implementation choice led to a fair bit of pain because the language would change so rapidly it was really unreasonable to expect the debugger team could keep track.

8/ Regardless, the EE for C# was implemented in mcee_cs.dll, and provided fairly basic evaluation. The joke on the team was that the EE had only a single error message which was 'managed EE does not understand expression's syntax.'

9/ TBH, it wasn’t far from the truth and unfortunately, it made the debugger significantly less powerful than it could have been as hover, evaluation in data windows, the immediate window, etc. were all impacted. The debugger also had this irritating property of evaluating

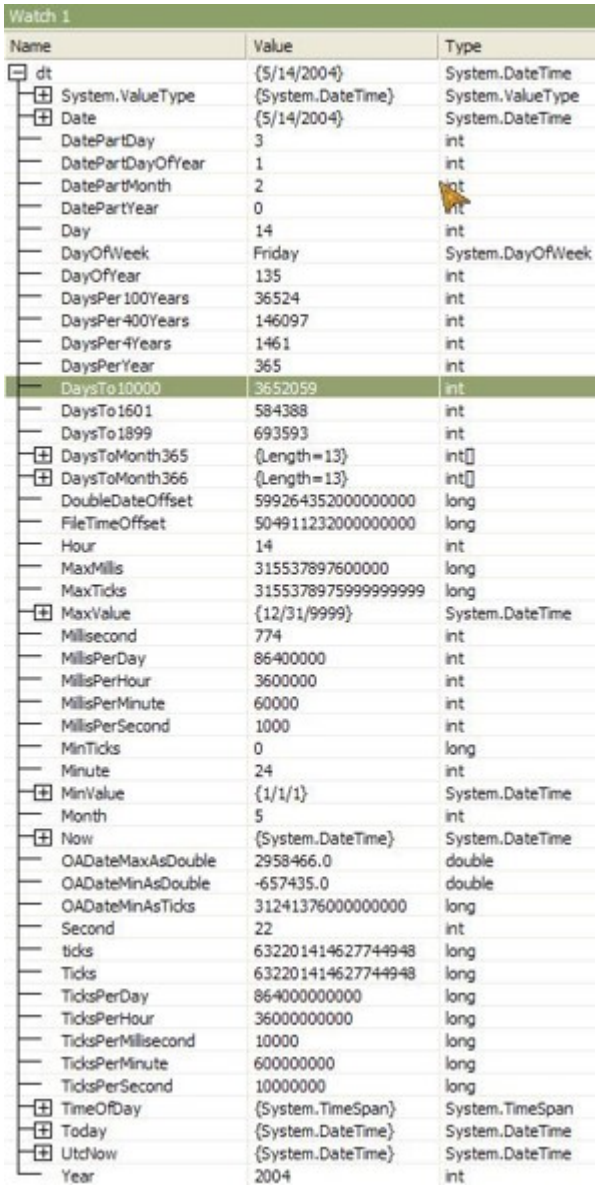
10/ expressions multiple times when refreshing UI, so if your evaluation happened to cause side-effects, your debugging state became untenable. That’s probably a separate thread though, so let’s just say, that we knew this was a problem but didn’t have time to address it for VS

11/ 2002. We started working on a new EE that was based on the *compiler* code base quickly thereafter, though it didn’t ship until VS 2005. We chose the compiler codebase versus the language service codebase, as we wanted it to be as faithful to the language as possible.

12/ We didn’t really have the architecture to treat the compiler as a service as that point in time, so the way this was done was by introducing a series of #ifdefs within the compiler codebase that implemented EE specific functionality.

13/ This did complicate the codebase, but enabled code-sharing. We produced a new csee.dll that was essentially the compiler with a different #define enabled on build. I am proud of the customer experience we achieved in VS 2005 with this change as it drastically improved the

14/ debugger experience both in terms of visualization and dynamic inspection. I’ll dig into this later, but just as an example, here is what the VS 2002/2003 view of DateTime looked like in the debugger versus what it looked like in VS 2005.



Watch 1

Name	Value	Type
dt	{5/14/2004}	System.DateTime
System.ValueType	{System.DateTime}	System.ValueType
Date	{5/14/2004}	System.DateTime
DatePartDay	3	int
DatePartDayOfYear	1	int
DatePartMonth	2	int
DatePartYear	0	int
Day	14	int
DayOfWeek	Friday	System.DayOfWeek
DayOfYear	135	int
DaysPer100Years	36524	int
DaysPer400Years	146097	int
DaysPer4Years	1461	int
DaysPerYear	365	int
DaysTo10000	3652059	int
DaysTo1601	584388	int
DaysTo1899	693593	int
DaysToMonth365	{Length=13}	int[]
DaysToMonth366	{Length=13}	int[]
DoubleDateOffset	599264352000000000	long
FileTimeOffset	504911232000000000	long
Hour	14	int
MaxMillis	315537897600000	long
MaxTicks	3155378975999999999	long
MaxValue	{12/31/9999}	System.DateTime
Millisecond	774	int
MillisPerDay	86400000	int
MillisPerHour	3600000	int
MillisPerMinute	60000	int
MillisPerSecond	1000	int
MinTicks	0	long
Minute	24	int
MinValue	{1/1/1}	System.DateTime
Month	5	int
Now	{System.DateTime}	System.DateTime
OADateMaxAsDouble	2958466.0	double
OADateMinAsDouble	-657435.0	double
OADateMinAsTicks	31241376000000000	long
Second	22	int
ticks	632201414627744948	long
Ticks	632201414627744948	long
TicksPerDay	864000000000	long
TicksPerHour	36000000000	long
TicksPerMillisecond	10000	long
TicksPerMinute	600000000	long
TicksPerSecond	10000000	long
TimeOfDay	{System.TimeSpan}	System.TimeSpan
Today	{System.DateTime}	System.DateTime
UtcNow	{System.DateTime}	System.DateTime
Year	2004	int