



---

## Debugger

Gary Taylor : Systems Architect

# Agenda

---

- Recent Changes
- Useful Features
- Manipulating data
- Mimicking your Runtime
  - External Libraries
  - What UI do you use?
- ATT
- Performance

# Some background

---

- The AB Suite Debugger is built into Visual Studio

# Some background

- The AB Suite Debugger is built into Visual Studio

The screenshot shows the Visual Studio IDE with the AB Suite Debugger. The main editor displays the following code:

```
1 ForEach arrayElement In ArrayExample
2
3     Add 1 COUNT1
4     arrayElement.aLongString := "ABCDEFGHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"
5     arrayElement.aNumber := 123456789012345
6     arrayElement.aString := "@"
7     arrayElement.agroup.anotherNumber := 1234567890
8     arrayElement.agroup.anotherString := "hello"
9     ArrayExample[count1] := arrayElement
10
11 End
12 Message Attention "Populated Array"
13
```

The Watch window shows the following variables:

Name	Value	Type
ArrayE...		P...
count1	0000...	N...
ArrayE...		R...
aLo...	"AB...	S...
aN...	1234...	N...
aStr...	"@"	S...
aGr...		R...

The Immediate Window shows the current state of the 'arrayexample' object:

```
arrayexample
ABCDEFHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890
lo 1234567890ABCDEFHJKLMNOPQRSTUVWXYZabcde
123456789012345@hello 123456789
4567890 123456789012345@he
XYZabcdefghijklmnopqrstuvwxyz1234567890
234567890ABCDEFHJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz12
12345@hello 1234567890ABCDEFHJKLMNOPQRSTU
123456789012345@hello
```

The Class View shows the following project structure:

- Report2
- STARTTEST
- TestArrays
- TestCall
- TestCase
- TestCP
- TestCP2
- TestList
- TestLock
- testPrint
- TestRecover
- TestRep

# Some background

---

- The AB Suite Debugger is built into Visual Studio

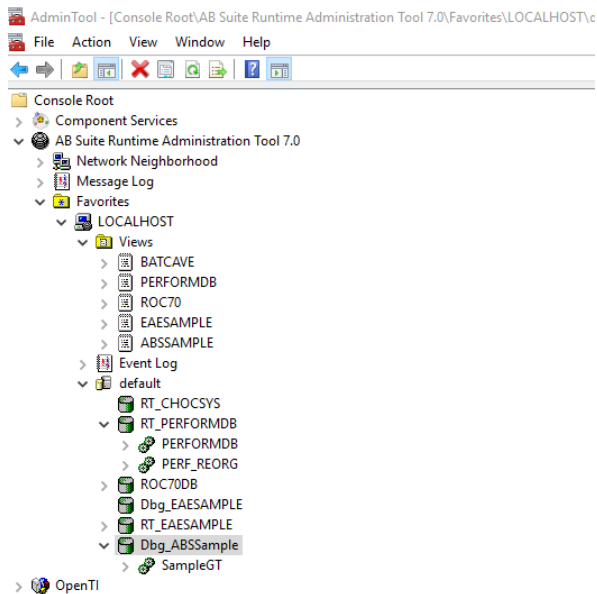
# Some background

---

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL

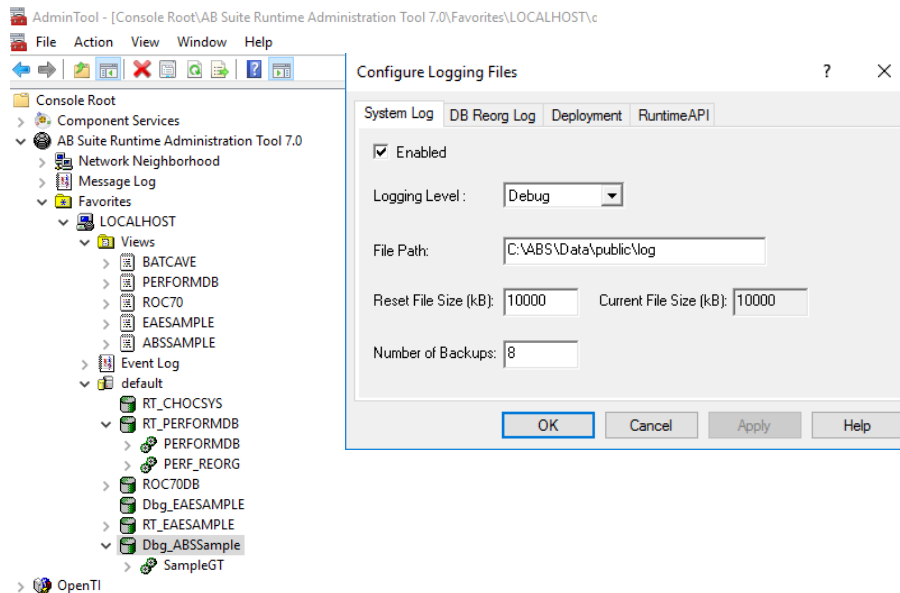
# Some background

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL



# Some background

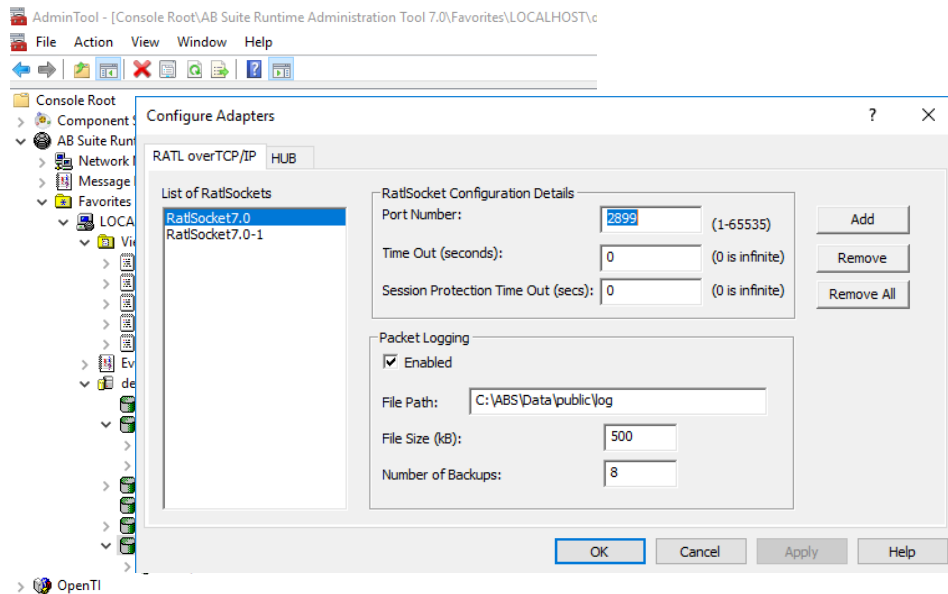
- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL





# Some background

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL



# Some background

---

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL
    - Makes use of the same WinForm Client
  - Has the same DB structure
    - Same Reorg process
    - Same Database Security concepts

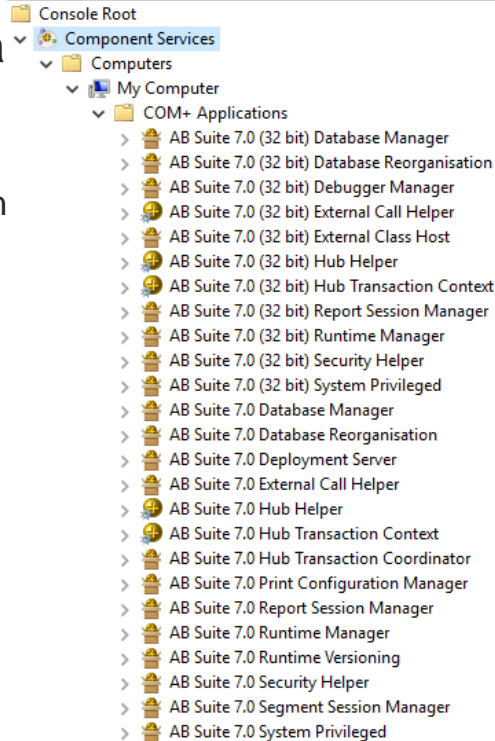
# Some background

---

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL
    - Makes use of the same WinForm Client
  - Has the same DB structure
    - Same Reorg process
    - Same Database Security concepts
- But it is also different
  - 32 bit not 64 bit

# Some background

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares the same tools
  - Utilises some of the same tools
    - Administration Client, Logging, RATL
    - Makes use of the same WinForm Client
  - Has the same DB structure
    - Same Reorg process
    - Same Database Security concepts
- But it is also different
  - 32 bit not 64 bit



Windows Runtime

Debugger

Windows Runtime

# Some background

---

- The AB Suite Debugger is built into Visual Studio
- Provides a virtual runtime that shares much with the AB Suite Windows Runtime
  - Utilises some of the same tools
    - Administration Client, Logging, RATL
    - Makes use of the same WinForm Client
  - Has the same DB structure
    - Same Reorg process
    - Same Database Security concepts
- But it is also different
  - 32 bit not 64 bit
  - C++ based not .Net
  - Interpreted
  - Wrappers Built to intercept calls from common UI



## Recent Changes

# Recent Changes - 1

---

- Quick Start debugging
  - Minimum debugger values auto populated
  - Default Suffix based on initials appended to Key values
    - Schema Name
    - Alternate Name
    - Database Name

# Recent Changes - 1

- Quick Start debugging
  - Minimum debugger version
  - Default Suffix based
    - Schema Name
    - Alternate Name
    - Database Name

>	<b>Client</b>	
▼	<b>Installation</b>	
	Debug System Name suffix	<b>GT</b>
▼	<b>Misc</b>	
	Value of Glb.Machine	As Per Host
▼	<b>Test Database</b>	
	Enable Host Database Access	False
	Database Schema Name	<b>SampleGT</b>
	Alternate Name	<b>SampleGT</b>
	Database Server Registration	<b>default</b>
	Database Name	<b>SampleGTDB</b>
	Reorganize Database	Yes
	Allow Recovery from Failed Reorganization	Yes
▼	<b>Test RSS Properties</b>	
	Enable Remote Call	False



# Recent Changes - 1

- Quick Start debugging
  - Minimum debugger values auto populated
  - Default Suffix based on initials appended to Key values
    - Schema Name
    - Alternate Name
    - Database Name
  - Will create the Debugger Database automatically if required
    - Does require “default” database registration
  - Can still configure values manually

>	<b>Client</b>	
▼	<b>Installation</b>	
	Debug System Name suffix	GT
▼	<b>Misc</b>	
	Value of Glb.Machine	As Per Host
▼	<b>Test Database</b>	
	Enable Host Database Access	False
	Database Schema Name	SampleGT
	Alternate Name	SampleGT
	Database Server Registration	default
	Database Name	SampleGTDB
	Reorganize Database	Yes
	Allow Recovery from Failed Reorganization	Yes
▼	<b>Test RSS Properties</b>	
	Enable Remote Call	False

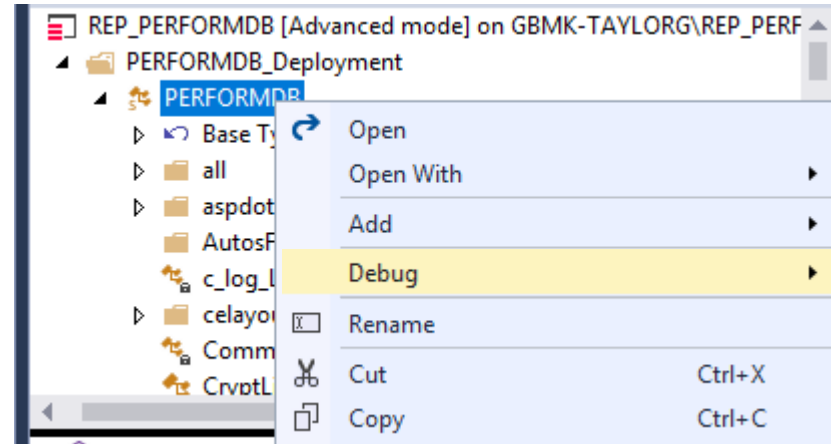
# Recent Changes - 2

---

- Debug what is selected in Class view
  - To debug Segment, select the Segment
  - To debug a Report, select the Report
  - To debug a Public Segment Method, select the Method

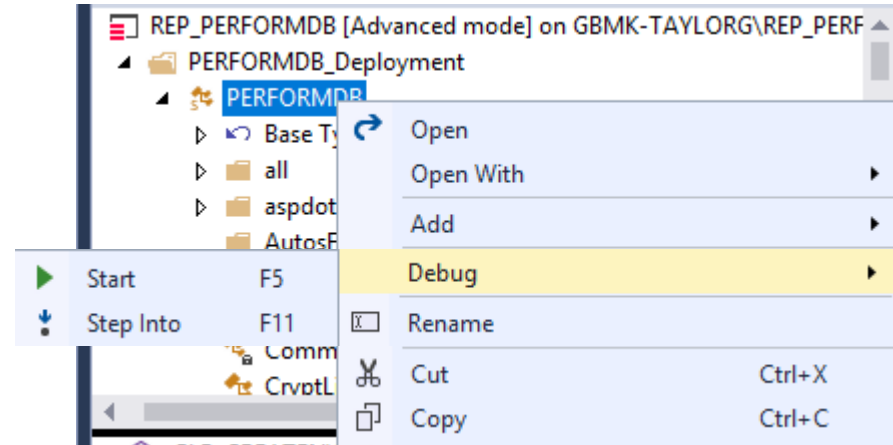
# Recent Changes - 2

- Debug what is selected in Class view
  - To debug Segment, select the Segment
  - To debug a Report, select the Report
  - To debug a Public Segment Method, select the Method



# Recent Changes - 2

- Debug what is selected in Class view
  - To debug Segment, select the Segment
  - To debug a Report, select the Report
  - To debug a Public Segment Method, select the Method
- Option to:
  - Step into (F11) - stops at first logic line
    - Pre 7.0 behaviour
  - Start (F5) - continues until first break point



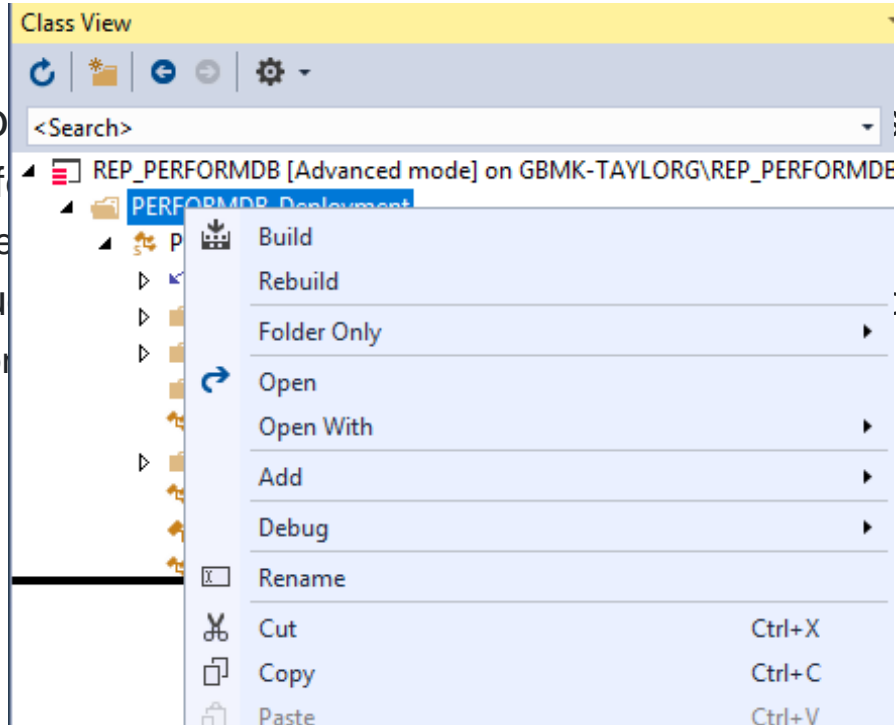
# Recent Changes - 3

---

- Debugger runs based on action rather than Configuration setting
  - Reduces need for separate Debugger Configuration
    - No Debug mode at the Configuration level
  - “Build” or “Debug” from same configuration depending on “Action”
    - Right click actions available in Class view

# Recent Changes - 3

- Debugger runs build
- Reduces need for  
• No Debug mode
- “Build” or “Debug”  
• Right click action



setting

on”

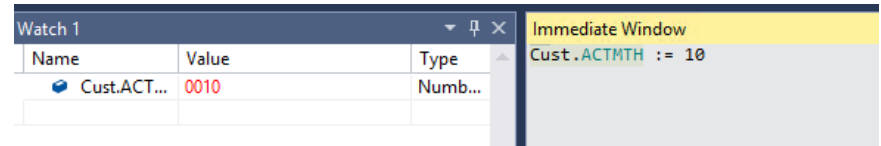
# Recent Changes - 3

---

- Debugger runs based on action rather than Configuration setting
  - Reduces need for separate Debugger Configuration
    - No Debug mode at the Configuration level
  - “Build” or “Debug” from same configuration depending on “Action”
    - Right click actions available in Class view
    - For example would allow “Build” of system from same Configuration as used with debugger
      - Wrappers automatically built as required no need to manual build

# Recent Changes - 3

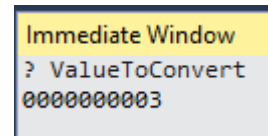
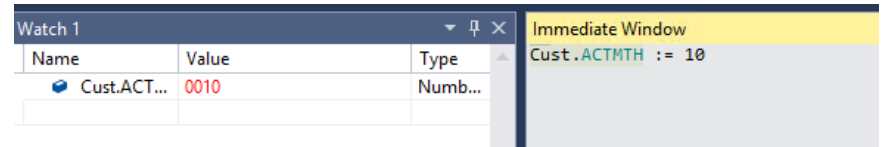
- Debugger runs based on action rather than Configuration setting
  - Reduces need for separate Debugger Configuration
    - No Debug mode at the Configuration level
  - “Build” or “Debug” from same configuration depending on “Action”
    - Right click actions available in Class view
    - For example would allow “Build” of system from same Configuration as used with debugger
      - Wrappers automatically built as required no need to manual build
- New Immediate Window
  - Allows you to Evaluate expressions
  - Edit Attribute values via use of Assignment clause :=





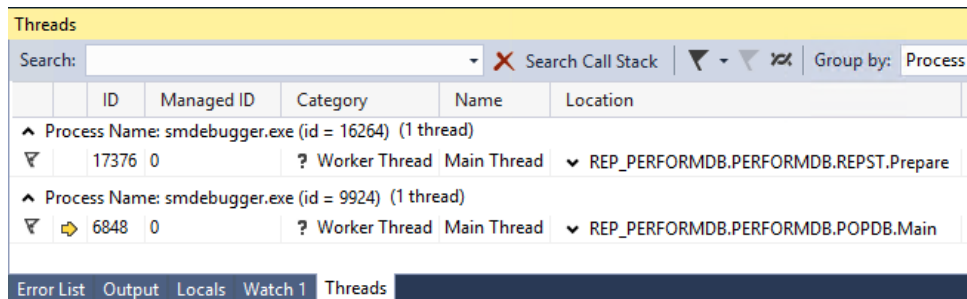
# Recent Changes - 3

- Debugger runs based on action rather than Configuration setting
  - Reduces need for separate Debugger Configuration
    - No Debug mode at the Configuration level
  - “Build” or “Debug” from same configuration depending on “Action”
    - Right click actions available in Class view
    - For example would allow “Build” of system from same Configuration as used with debugger
      - Wrappers automatically built as required no need to manual build
- New Immediate Window
  - Allows you to Evaluate expressions
  - Edit Attribute values via use of Assignment clause :=
  - Print values via Debug.Print (? Is shortform)



# Recent Changes - 4

- Reports now run in same copy of Visual Studio
  - No longer starts a second Visual Studio instance when starting Reports from Online
  - Report runs as Separate Thread

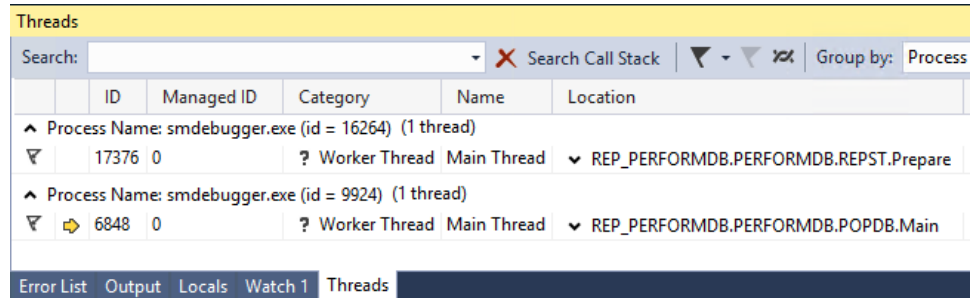


The screenshot shows the 'Threads' window in Visual Studio. It features a search bar, a 'Search Call Stack' button, and a 'Group by: Process' dropdown. The table below lists threads for two instances of smdebugger.exe.

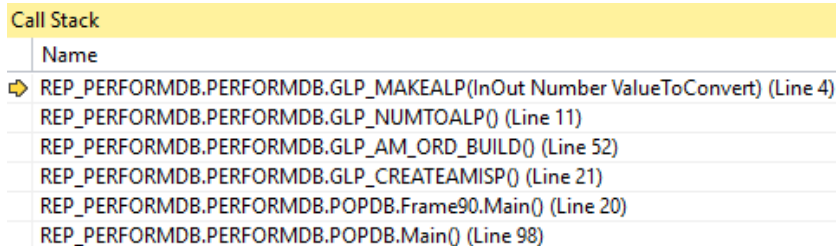
	ID	Managed ID	Category	Name	Location
^ Process Name: smdebugger.exe (id = 16264) (1 thread)					
▼	17376	0	? Worker Thread	Main Thread	▼ REP_PERFORMDB.PERFORMDB.REPST.Prepare
^ Process Name: smdebugger.exe (id = 9924) (1 thread)					
▼	6848	0	? Worker Thread	Main Thread	▼ REP_PERFORMDB.PERFORMDB.POPDB.Main

# Recent Changes - 4

- Reports now run in same copy of Visual Studio
  - No longer starts a second Visual Studio instance when starting Reports from Online
  - Report runs as Separate Thread



- Call Stack Window displays Methods and parameters

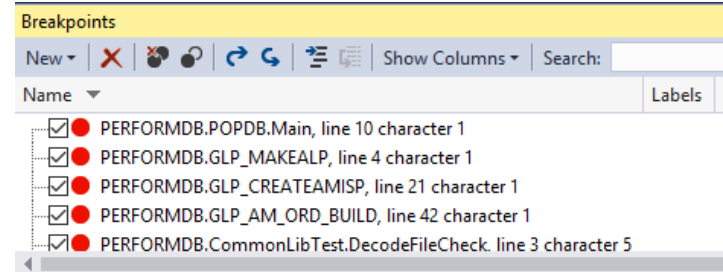




## Useful Features

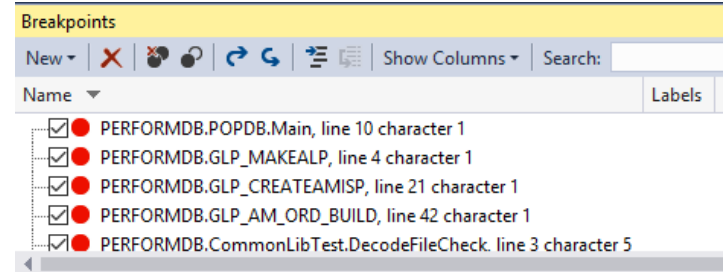
# Useful features - 1

- Breakpoints
  - Breakpoint Window
    - Allows you enable or disable any breakpoint
    - Quick way to locate and go to any breakpoint



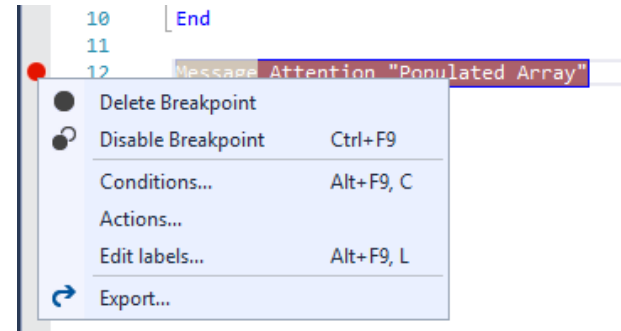
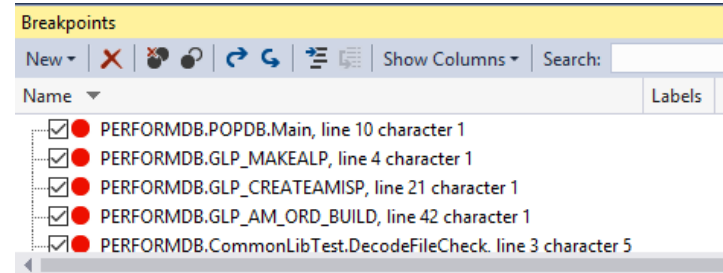
# Useful features - 1

- Breakpoints
  - Breakpoint Window
    - Allows you enable or disable any breakpoint
    - Quick way to locate and go to any breakpoint
  - Conditional break points



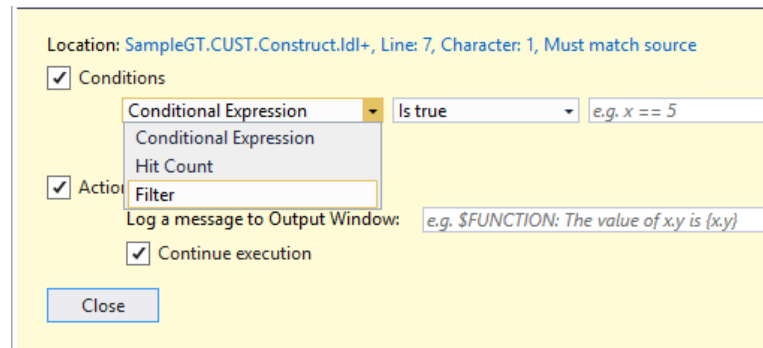
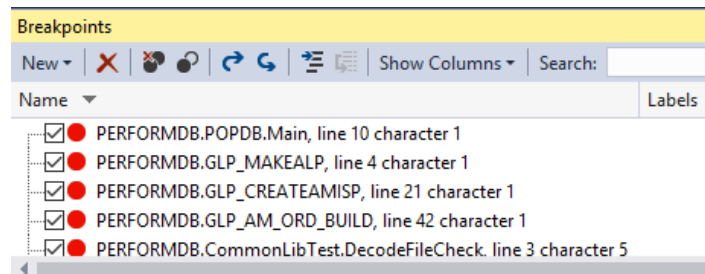
# Useful features - 1

- Breakpoints
  - Breakpoint Window
    - Allows you enable or disable any breakpoint
    - Quick way to locate and go to any breakpoint
  - Conditional break points



# Useful features - 1

- Breakpoints
  - Breakpoint Window
    - Allows you enable or disable any breakpoint
    - Quick way to locate and go to any breakpoint
  - Conditional break points
    - Break on:
      - Function – when reach line
      - Hit – When that break point has been executed “hit” times
      - Conditional – when defined condition comes true
    - Options to do something or just halt





# Useful features - 2

---

- LOG LDL+ Verb
  - Provides ability to Log messages to Debugger output window
    - For Windows Runtime Logs to the System log

```
LOG { DEBUG | RELEASE | ALWAYS } [ ERROR | WARNING | HALT ] expression [ expression ]
```

# Useful features - 2

---

- LOG LDL+ Verb
  - Provides ability to Log messages to Debugger output window

- For Windows Runtime Logs to the System log

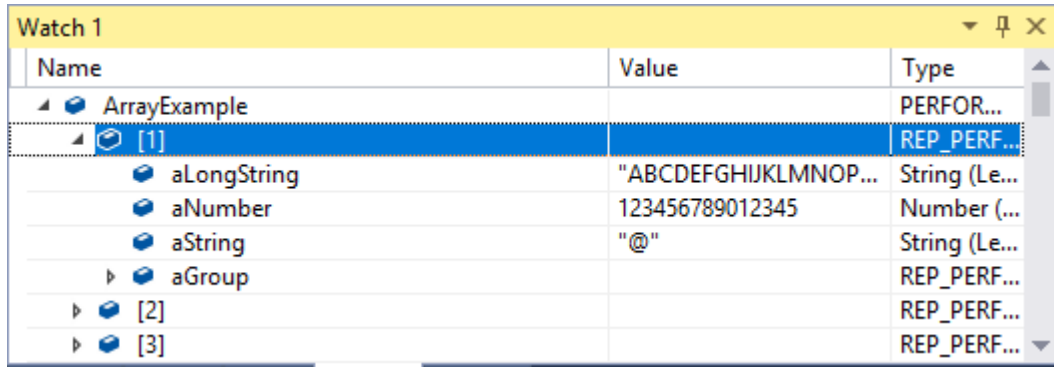
```
LOG { DEBUG | RELEASE | ALWAYS } [ ERROR | WARNING | HALT ] expression [ expression ]
```

- No Op on MCP

# Useful features - 2

- LOG LDL+ Verb
  - Provides ability to Log messages to Debugger output window
    - For Windows Runtime Logs to the System log
- Ability to monitor individual elements in an array

```
LOG { DEBUG | RELEASE | ALWAYS } [ ERROR | WARNING | HALT ] expression [ expression ]
```



The screenshot shows a debugger's Watch window titled "Watch 1". It contains a table with three columns: Name, Value, and Type. The table lists the following items:

Name	Value	Type
ArrayExample		PERFOR...
[1]		REP_PERF...
aLongString	"ABCDEFGHJKLMNOP..."	String (Le...
aNumber	123456789012345	Number (...)
aString	"@"	String (Le...
aGroup		REP_PERF...
[2]		REP_PERF...
[3]		REP_PERF...

# Useful features - 2

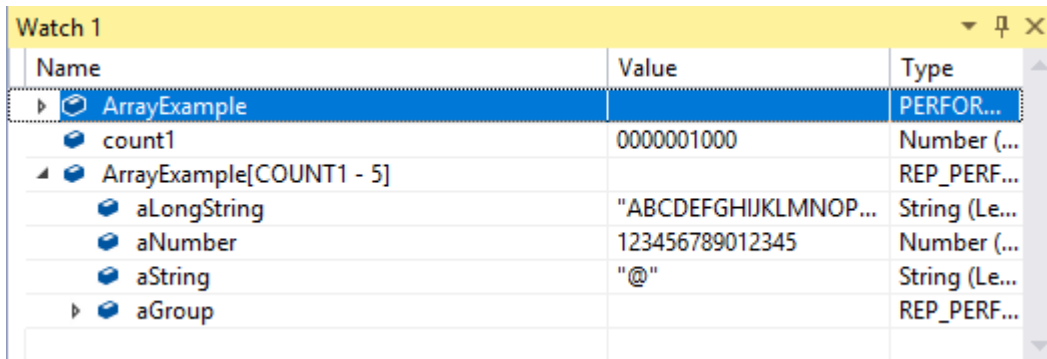
- LOG LDL+ Verb
  - Provides ability to Log messages to Debugger output window

- For Windows Runtime Logs to the System log

```
LOG { DEBUG | RELEASE | ALWAYS } [ ERROR | WARNING | HALT ] expression [ expression ]
```

- No Op on MCP

- Ability to monitor individual elements in an array
  - Capable of using expression to derive index value



Name	Value	Type
▶ ArrayExample		PERFOR...
count1	0000001000	Number (...)
▲ ArrayExample[COUNT1 - 5]		REP_PERF...
aLongString	"ABCDEFGHJKLMNOP..."	String (Le...)
aNumber	123456789012345	Number (...)
aString	"@"	String (Le...)
▶ aGroup		REP_PERF...

# Useful features - 2

- LOG LDL+ Verb

- Provides ability to Log messages to Debugger output window

- For Windows Runtime Logs to the System log

```
LOG { DEBUG | RELEASE | ALWAYS } [ ERROR | WARNING | HALT ] expression [ expression ]
```

- No Op on MCP

- Ability to monitor individual elements in an array

- Capable of using expression to derive index value

- Display whole array as “string” moved to the new “Immediate” window

Name	Value	Type
ArrayExample		PERF...
count1	0000001000	Num...
ArrayExample[COUNT1 - 5]		REP...
aLongString	"ABCDEFGHJK..."	Strin...
aNumber	1234567890123...	Num...
aString	"@"	Strin...
aGroup		REP...

```
arrayexample
ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890
lo
1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890
123456789012345@hello
123456789012345@hello
123456789012345@hello
XYZabcdefghijklmnopqrstvwxyz1234567890
12345678901234567890
234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890
12345@hello
1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890
123456789012345@hello
123456789012345@hello
```

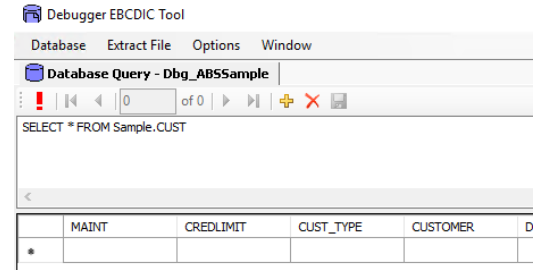


---

## Manipulating Data

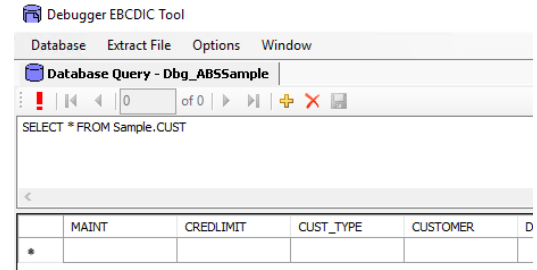
# Manipulating data

- Solution platform determines how data stored by Debugger
  - Platform MCP - data (DB and Extract files) in EBCDIC
    - Data returned in same collating sequence as on MCP host
    - EBCDIC Tool supplied to help view and manipulate data
    - Allows record updating but not bulk load
  - For other platforms data stored in ASCII
    - Use standard windows tools to manipulate data e.g. SQL Server Management Studio (SSMS)



# Manipulating data

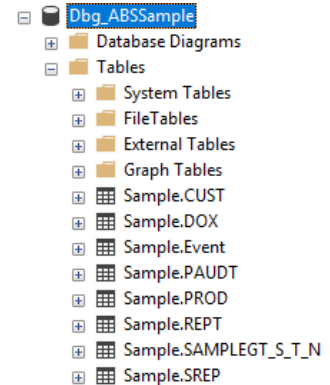
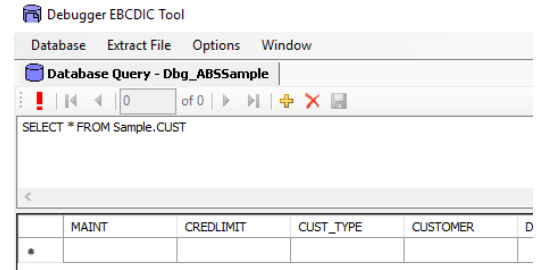
- Solution platform determines how data stored by Debugger
  - Platform MCP - data (DB and Extract files) in EBCDIC
    - Data returned in same collating sequence as on MCP host
    - EBCDIC Tool supplied to help view and manipulate data
    - Allows record updating but not bulk load
  - For other platforms data stored in ASCII
    - Use standard windows tools to manipulate data e.g. SQL Server Management Studio (SSMS)
- Populating Debugger databases





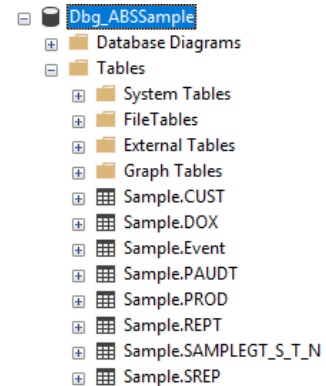
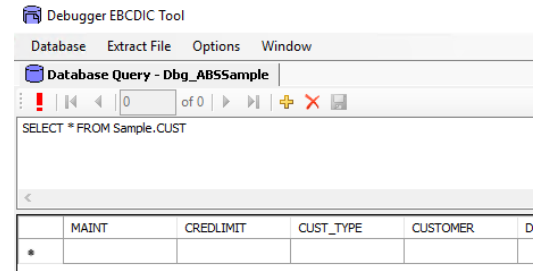
# Manipulating data

- Solution platform determines how data stored by Debugger
  - Platform MCP - data (DB and Extract files) in EBCDIC
    - Data returned in same collating sequence as on MCP host
    - EBCDIC Tool supplied to help view and manipulate data
    - Allows record updating but not bulk load
  - For other platforms data stored in ASCII
    - Use standard windows tools to manipulate data e.g. SQL Server Management Studio (SSMS)
- Populating Debugger databases
  - Database Schema matches Windows Runtime
    - Easy to Clone Windows Runtime Databases as Debugger Databases
    - SSMS can be used to load ASCII data via simple SQL statements



# Manipulating data

- Solution platform determines how data stored by Debugger
  - Platform MCP - data (DB and Extract files) in EBCDIC
    - Data returned in same collating sequence as on MCP host
    - EBCDIC Tool supplied to help view and manipulate data
    - Allows record updating but not bulk load
  - For other platforms data stored in ASCII
    - Use standard windows tools to manipulate data e.g. SQL Server Management Studio (SSMS)
- Populating Debugger databases
  - Database Schema matches Windows Runtime
    - Easy to Clone Windows Runtime Databases as Debugger Databases
    - SSMS can be used to load ASCII data via simple SQL statements
  - Different Schema definition to MCP
    - Most likely need to load data from another source
    - Need to convert data back to EBCDIC for MCP Debugger Target



# Populating Debugger Databases – SSIS Generator

- **UK Developed tool** designed to help transfer data between different source/targets
  - Auto Generates SSIS Packages to move data
  - Understands different platform Schema definitions
  - Any combination of Source and Destination allowed

Source Data Type	Target Destinations
SQL Server (EAE or AB Suite)	SQL Server (AB Suite) Optional EBCDIC
MCP (EAE/AB Suite)	MCP (EAE/AB Suite)
Oracle (EAE)	EAE Dev Test
OS2200	Flat Files (CSV)
EAE Developer Test	

AB Suite SSIS Package Generator

File Settings Help

Table list file C:\Temp\SSISOut\Tablelist.txt ... Generate

Output SSIS Package folder C:\Temp\SSISOut ...

**System Details**

Truncate target Tables?  Migrate GLB\_DTIME?  
 Remove Stored procedure at end?

**Source connection details**

Source DB is AB Suite?

Source Type  SQL Server  Oracle  OS2200  MCP  EAD

Host Name GBMK-TAYLORG

Database P1  64bit?

UserName UserName Password \*\*\*\*\*  NXFile ?

EAE DB Name ...

**Destination connection details**

Destination Type  SQL Server  MCP  Flat File  EAD  DB Schema

Host name GBMK-TAYLORG2 ...

Database RT\_DESTDB  EBCDIC?

UserName UserName Password \*\*\*\*\*

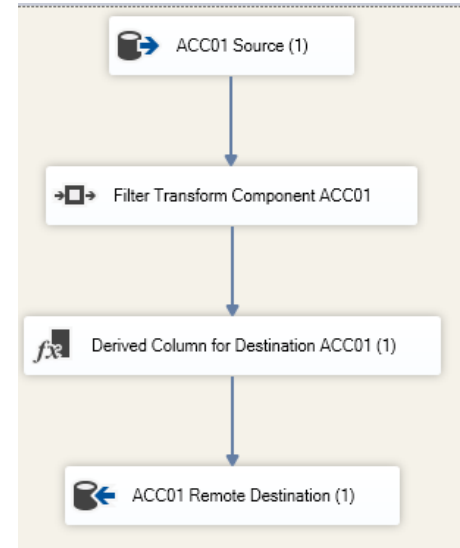
Schema Name DESTSCHEMA  De-Dup?

Go Cancel Exit

» [Read More](#) (July 2019 Developing Agility article)

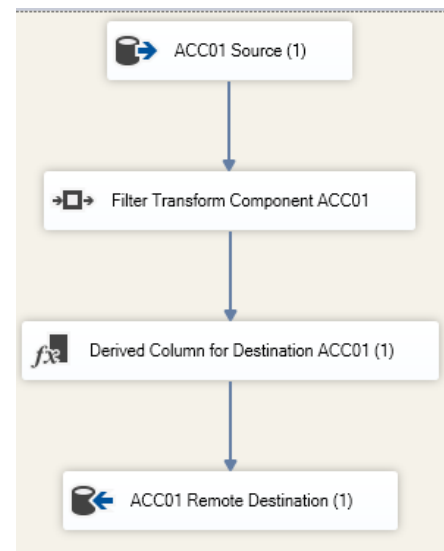
# Populating Debugger Databases – SSIS Generator - 2

- Generator builds standard SSIS Packages
  - Can be edited via Visual Studio (SQL Data Tools)
  - One table per Package (.dtsx)
  - Uses custom Transforms and Connectors for some platforms and capabilities



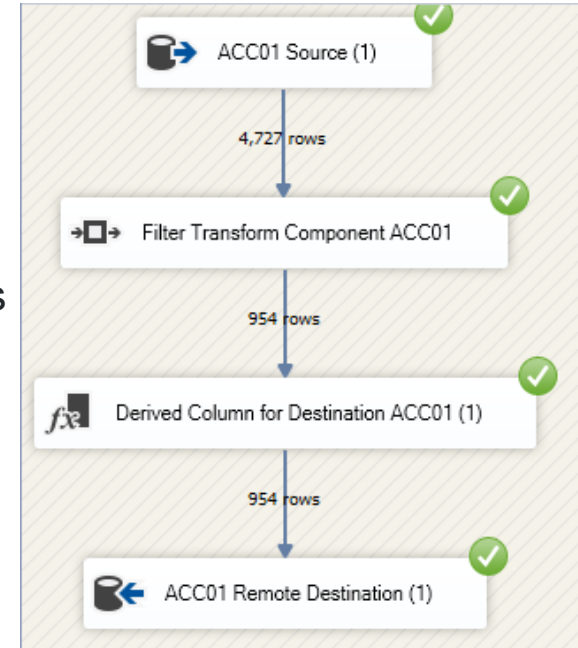
# Populating Debugger Databases – SSIS Generator - 2

- Generator builds standard SSIS Packages
  - Can be edited via Visual Studio (SQL Data Tools)
  - One table per Package (.dtsx)
  - Uses custom Transforms and Connectors for some platforms and capabilities
    - **Recently added capability to Filter rows**
      - Use simple rules to determine what rows are sent to target database
      - E.g. `SUBSTR(SUB_ACC_NO,1,5) = "A1234"`



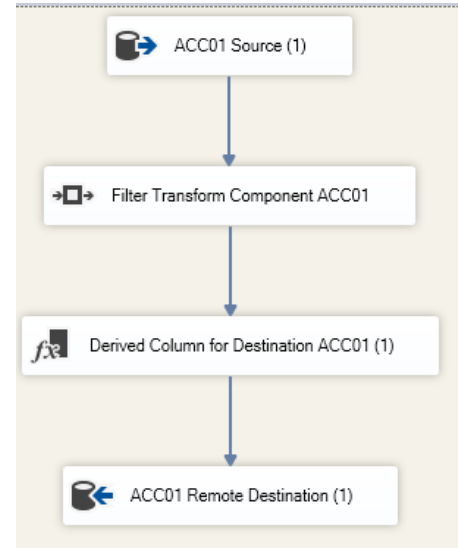
# Populating Debugger Databases – SSIS Generator - 2

- Generator builds standard SSIS Packages
  - Can be edited via Visual Studio (SQL Data Tools)
  - One table per Package (.dtsx)
  - Uses custom Transforms and Connectors for some platforms and capabilities
    - Recently added capability to Filter rows
      - Use simple rules to determine what rows are sent to target database
      - E.g. `SUBSTR(SUB_ACC_NO,1,5) = "A1234"`



# Populating Debugger Databases – SSIS Generator - 2

- Generator builds standard SSIS Packages
  - Can be edited via Visual Studio (SQL Data Tools)
  - One table per Package (.dtsx)
  - Uses custom Transforms and Connectors for some platforms and capabilities
    - Recently added capability to Filter rows
      - Use simple rules to determine what rows are sent to target database
      - E.g. `SUBSTR(SUB_ACC_NO,1,5) = "A1234"`
- Can run packages manually or via script
  - Script keeps constant number of packages running
  - Retries any failures



```
Administrator: C:\WINDOWS\system32\cmd.exe
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

12/10/2020 11:48:00 -- Logfile RunParallelSSISPkgs.log Created
12/10/2020 11:48:00 -- Parallel SSIS package runner : version 1.8
12/10/2020 11:48:00 -- Written by Gary Taylor of Unisys UK Ltd, gary.j.taylor@unisys.com
12/10/2020 11:48:00 --
12/10/2020 11:48:00 -- MaxThreads = 7
12/10/2020 11:48:00 -- SSISPkgs001_ACC01.dtsx Started Processing
12/10/2020 11:48:00 -- SSISPkgs002_ACC02.dtsx Started Processing
12/10/2020 11:48:01 -- SSISPkgs003_ACC04.dtsx Started Processing
12/10/2020 11:48:01 --
12/10/2020 11:48:01 -- All Packages started, awaiting for them to complete
12/10/2020 11:48:01 --
```

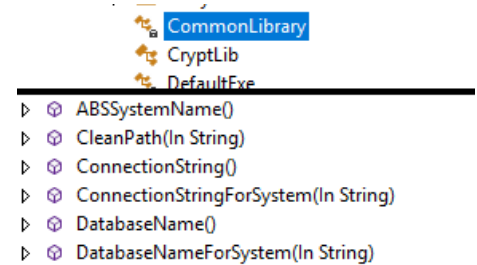


## Mimicking your Runtime



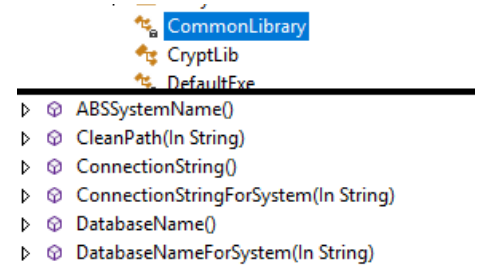
# External Libraries – Call()

- Possible to use external Libraries with Debugger
  - Access MCP Host based libraries via LRSS
  - Access Windows Libraries (All target platforms)



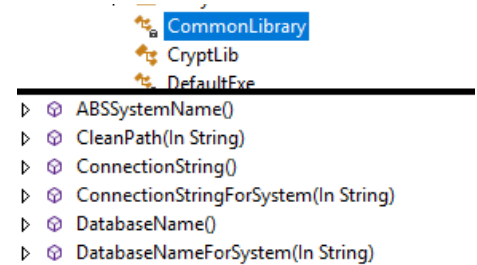
# External Libraries – Call()

- Possible to use external Libraries with Debugger
  - Access MCP Host based libraries via LRSS
  - Access Windows Libraries (All target platforms)
- Accessing .Net Libraries
  - Implementation is different to Windows Runtime
    - COM versus COM+ : so needs to be registered



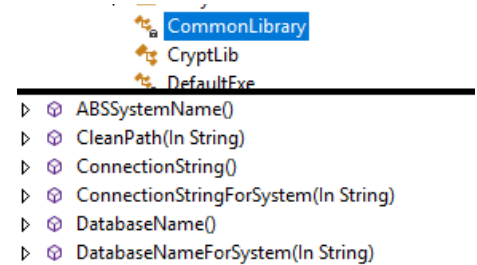
# External Libraries – Call()

- Possible to use external Libraries with Debugger
  - Access MCP Host based libraries via LRSS
  - Access Windows Libraries (All target platforms)
- Accessing .Net Libraries
  - Implementation is different to Windows Runtime
    - COM versus COM+ : so needs to be registered
  - Can detect running in Debugger



# External Libraries – Call()

- Possible to use external Libraries with Debugger
  - Access MCP Host based libraries via LRSS
  - Access Windows Libraries (All target platforms)
- Accessing .Net Libraries
  - Implementation is different to Windows Runtime
    - COM versus COM+ : so needs to be registered
  - Can detect running in Debugger



```
' AB Suite 7.0 runtime is 64Bit to call 64bit Libraries from Debugger uses a different executable  
' Change to reflect this includes making he variable global so we can reference elsewhere in the code  
If Environment.CommandLine.ToUpper.Contains("DEBUGGER") Then IsDebugger = True
```

# External Libraries – Call()

- Possible to use external Libraries with Debugger

- Access MCP Host based libraries via LRSS
- Access Windows Libraries (All target platforms)

- Accessing .Net Libraries

- Implementation is different to Windows Runtime

- COM versus COM+ : so needs to be registered

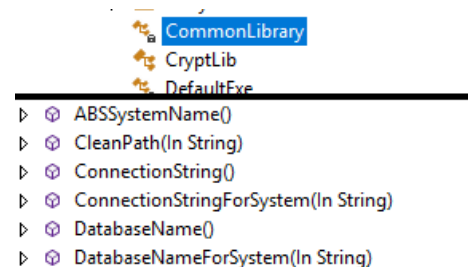
- Can detect running in Debugger

```
' AB Suite 7.0 runtime is 64Bit to call 64bit Libraries from Debugger uses a different executable  
' Change to reflect this includes making the variable global so we can reference elsewhere in the code  
If Environment.CommandLine.ToUpper.Contains("DEBUGGER") Then IsDebugger = True
```

- No Application folder - Runs from Developer\Bin installation folder

- From Experience:

- Utilise a local folder per developer – We have used `%HOMEDRIVE%%HOMEPATH%\SMDEBUGGER`
- Use Parameter files to define “Debugger System” details
- Call AB Suite COM+ Services to get further details – Debugger system information stored in NGRuntime.xml



# What UI do use for Testing?

---

- WinForm Client by Default

# What UI do use for Testing?

---

- WinForm Client by Default
- Possible to use own UI

# What UI do use for Testing?

- WinForm Client by Default
- Possible to use own UI
  - Set option to start RATL Server
  - Connect via Component Enabler
    - Segment and Alternate name must match
    - Define VIEW via Admin Tool
  - Change in debug configuration allows Bundle Builds

The image shows two screenshots related to the testing environment. The top screenshot is the 'REP\_PERFORMDB Property Pages' dialog box, showing configuration for a 'Debug' configuration on a 'Windows' platform. The 'Client' section is expanded, showing 'Application To Start' as 'C:\ABS\Bin64\RatlSocket1.exe', 'Working Directory' as 'C:\ABS\Bin64', and 'Debug System Name suffix' as 'TG'. The 'Installation' section shows 'Value of Glb.Machine' as 'As Per Host'. The bottom screenshot is the 'AB Suite Runtime Administration Tool 7.0' interface, showing a tree view of the system. The 'LOCALHOST' folder is expanded, showing 'Views' and 'default' folders. The 'default' folder is expanded, showing 'Dbg\_ABSSample' and 'SampleGT' folders. The 'SampleGT' folder is selected.

Test Database	
Database Schema Name	Sample
Alternate Name	SampleGT
Database Server Registration	default
Database Name	Dbg_ABSSample
...	..



# What UI do use for Testing?

- WinForm Client by Default
- Possible to use own UI
  - Set option to start RATL Server
  - Connect via Component Enabler
    - Segment and Alternate name must match
    - Define VIEW via Admin Tool
  - Change in debug configuration allows Bundle Builds
  - Opens up possibilities of using other testing tools

The image shows two screenshots related to the testing environment. The top screenshot is the 'REP\_PERFORMDB Property Pages' dialog box, showing configuration for a 'Debug' configuration on a 'Windows' platform. The 'Client' section is expanded, showing 'Application To Start' as 'C:\ABS\Bin64\RatlSocket1.exe', 'Working Directory' as 'C:\ABS\Bin64', and 'Debug System Name suffix' as 'TG'. The 'Installation' section shows 'Value of Glb.Machine' as 'As Per Host'. The bottom screenshot shows the 'AB Suite Runtime Administration Tool 7.0' interface. The 'Favorites' section is expanded to show 'LOCALHOST' > 'Views' > 'ABSSAMPLE' > 'default' > 'Dbg\_ABSSample' > 'SampleGT'. To the right of this interface is a table titled 'Test Database'.

Test Database	
Database Schema Name	Sample
Alternate Name	SampleGT
Database Server Registration	default
Database Name	Dbg_ABSSample
...	..

# What UI do use for Testing?

- WinForm Client by Default
- Possible to use own UI
  - Set option to start RATL Server
  - Connect via Component Enabler
    - Segment and Alternate name must match
    - Define VIEW via Admin Tool
  - Change in debug configuration allows Bundle Builds
  - Opens up possibilities of using other testing tools
    - Want to use tools like Selenium?

The image shows two screenshots from the AB Suite software. The top screenshot is the 'REP\_PERFORMDB Property Pages' dialog box, which is used for configuring the debug environment. It has a 'Configuration' dropdown set to 'Debug' and a 'Platform' dropdown set to 'Windows'. The 'Client' section is expanded, showing the 'Application To Start' as 'C:\ABS\Bin64\RatlSocket1.exe', 'Command Line Arguments' as empty, and 'Working Directory' as 'C:\ABS\Bin64'. The 'Installation' section shows 'Debug System Name suffix' as 'TG'. The 'Misc' section shows 'Value of Glb.Machine' as 'As Per Host'.

The bottom screenshot shows the 'AB Suite Runtime Administration Tool 7.0' interface. It has a tree view on the left with the following structure:

- AB Suite Runtime Administration Tool 7.0
  - Network Neighborhood
  - Message Log
  - Favorites
    - LOCALHOST
      - Views
        - ABSSAMPLE
        - default
          - Dbg\_ABSSample
            - SampleGT

On the right side of the screenshot, there is a table titled 'Test Database' with the following data:

Test Database	
Database Schema Name	Sample
Alternate Name	SampleGT
Database Server Registration	default
Database Name	Dbg_ABSSample
...	..

# What UI do use for Testing?

- WinForm Client by Default
- Possible to use own UI
  - Set option to start RATL Server
  - Connect via Component Enabler
    - Segment and Alternate name must match
    - Define VIEW via Admin Tool
  - Change in debug configuration allows Bundle Builds
  - Opens up possibilities of using other testing tools
    - Want to use tools like Selenium?
      - Using the Standard ASP.Net Generator would open options for any browser based testing tool.

The image shows two screenshots related to the AB Suite Runtime Administration Tool 7.0. The top screenshot is the 'REP\_PERFORMDB Property Pages' dialog box, which is used for configuring the debug environment. It has a 'Configuration' dropdown set to 'Debug' and a 'Platform' dropdown set to 'Windows'. The 'Client' section is expanded, showing the following properties:

Property	Value
Application To Start	C:\ABS\Bin64\RatlSocket1.exe
Command Line Arguments	
Working Directory	C:\ABS\Bin64
Installation	
Debug System Name suffix	TG
Misc	
Value of Glb.Machine	As Per Host

The bottom screenshot is a tree view of the 'AB Suite Runtime Administration Tool 7.0'. The tree structure is as follows:

- AB Suite Runtime Administration Tool 7.0
  - Network Neighborhood
  - Message Log
  - Favorites
    - LOCALHOST
      - Views
        - ABSSAMPLE
        - default
          - Dbg\_ABSSample
            - SampleGT

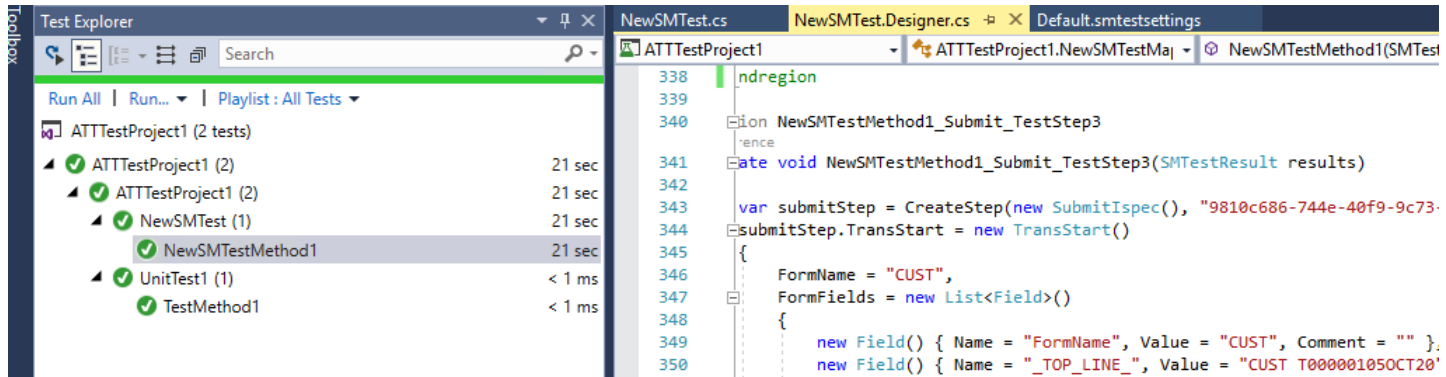
Test Database	
Database Schema Name	Sample
Alternate Name	SampleGT
Database Server Registration	default
Database Name	Dbg_ABSSample
...	..



ATT

# Create Automated Tests

- Using alternate UIs opens up use with AB Suite Automated Test Tool (ATT)
  - Get developers to create Test Scripts
    - Start with simple Record and Playback but ability to extend later
  - Test in Debugger
  - Automate as part of DevOps for automatic testing after a build
    - Create suites of test scripts as you build or change functionality



```
338 | ndregion
339 |
340 | [ion NewSMTestMethod1_Submit_TestStep3
341 |     'ence
342 |     ate void NewSMTestMethod1_Submit_TestStep3(SMTestResult results)
343 |     var submitStep = CreateStep(new SubmitSpec(), "9810c686-744e-40f9-9c73-
344 |     submitStep.TransStart = new TransStart()
345 |     {
346 |         FormName = "CUST",
347 |         FormFields = new List<Field>()
348 |         {
349 |             new Field() { Name = "FormName", Value = "CUST", Comment = "" }.
350 |             new Field() { Name = "_TOP_LINE_", Value = "CUST T00000105OCT20"
```



**Performance**

# Performance

---

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

# Performance

---

- Number of general performance improvements included with last few releases
  - » [Read More](#) – (Developing Agility Oct 2018)
  - But you can also help yourself:



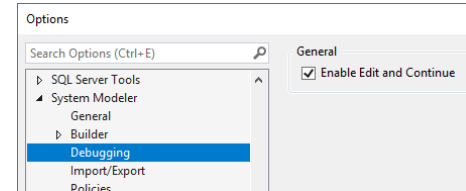
# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:

- Disabling Edit and Continue if you don't need to edit code – much easier in 7.0

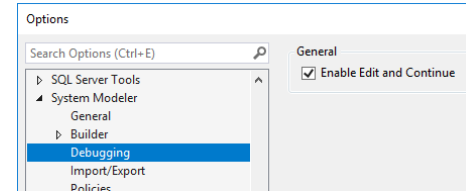


# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:
  - Disabling Edit and Continue if you don't need to edit code – much easier in 7.0
  - Ability to run to Break point in 7.0 (F5 - Start) rather than previous “Step Into”

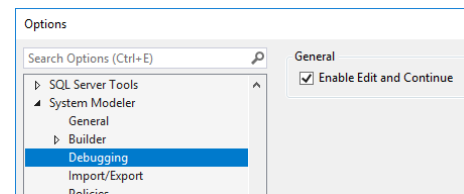


# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:
  - Disabling Edit and Continue if you don't need to edit code – much easier in 7.0
  - Ability to run to Break point in 7.0 (F5 - Start) rather than previous “Step Into”
  - Watch windows have to be updated so limit Windows and what is displayed
    - One reason Arrays not shown as strings in watch windows anymore

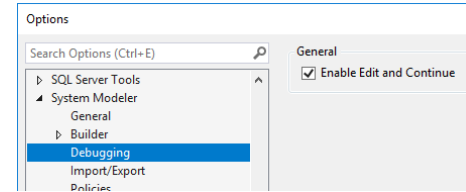


# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:
  - Disabling Edit and Continue if you don't need to edit code – much easier in 7.0
  - Ability to run to Break point in 7.0 (F5 - Start) rather than previous “Step Into”
  - Watch windows have to be updated so limit Windows and what is displayed
    - One reason Arrays not shown as strings in watch windows anymore
- Improve initial debug start up time by limiting what is built



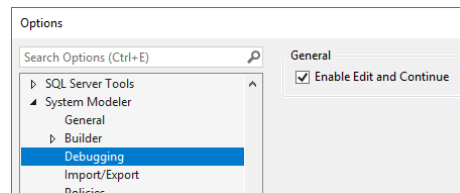
# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:

- Disabling Edit and Continue if you don't need to edit code – much easier in 7.0
- Ability to run to Break point in 7.0 (F5 - Start) rather than previous “Step Into”
- Watch windows have to be updated so limit Windows and what is displayed
  - One reason Arrays not shown as strings in watch windows anymore

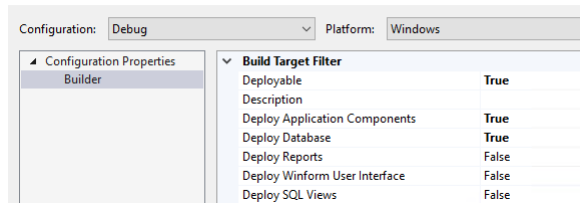


- Improve initial debug start up time by limiting what is built

- Before you can debug a “Wrapper” has to be built for the Segment and all Reports.

- Don't make Reports Deployable at the Top level deployment folder
- Use lower level Folder

PERFORMDB\_Deployment Property Pages



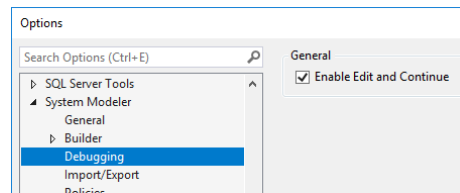
# Performance

- Number of general performance improvements included with last few releases

» [Read More](#) – (Developing Agility Oct 2018)

- But you can also help yourself:

- Disabling Edit and Continue if you don't need to edit code – much easier in 7.0
- Ability to run to Break point in 7.0 (F5 - Start) rather than previous “Step Into”
- Watch windows have to be updated so limit Windows and what is displayed
  - One reason Arrays not shown as strings in watch windows anymore



- Improve initial debug start up time by limiting what is built

- Before you can debug a “Wrapper” has to be built for the Segment and all Reports.

- Don't make Reports Deployable at the Top level deployment folder
- Use lower level Folder

- Don't Reorg if you don't Need too

PERFORMDB\_Deployment Property Pages

Configuration: Debug Platform: Windows

Configuration Properties	
Builder	
Test Database	
Database Schema Name	PERFORMDB
Alternate Name	PERFORMDBTG
Database Server Registration	default
Database Name	RT_PERFORMDB
Reorganize Database	Yes
Allow Recovery from Failed Reorganization	No
Enable User Maintained Tables	True

Build Target Filter	
Deployable	True
Description	
Deploy Application Components	True
Deploy Database	True
Deploy Reports	False
Deploy Winform User Interface	False
Deploy SQL Views	False

# Thank You

