

How To Create a Continuous Testing Environment with the TestComplete REST Api and Virtual Machines



Introduction

Stability and productivity are everything with automated testing. If you're going to be productive with TestComplete you must have a way to test and run your automated tests outside of your development machine. As your tests grow you cannot afford to waste time tying up your development machine as you run your automated tests. It's crucial that you have separate environments where you can execute tests independently.

We're going to walk you through a simple'ish setup where you develop on one machine and run on a separate virtual machine using TestExecute. Execute a script, your tests run on a VM, results available on your development machine.

What You'll Learn

When you follow this "how to" document you'll learn:

- i. How to configure a VM and install TestExecute
- ii. How TestComplete/TestExecute can work in 'shared' mode (sharing access to a projects files)
- iii. How to use the TestComplete API to automatically run your tests
- iv. How to capture and save test results in an Html format that you can access on a shared drive

Ultimately, you'll learn how to increase your productivity and increase the stability of your automated tests.

Stage 1 – Create Your TestComplete Project

We need to setup a development environment where you can develop tests. We'll refer to this as your main desktop/laptop machine. On this main machine we'll develop a simple automation script. We'll create a script that just opens Notepad, creates some content and then closes notepad.

1. Open TestComplete on your main machine, create a new project suite and create a new project. You can name them...

Project Suite => Notepad Automation

Project => Notepad1

You should have something like this:

TestComplete - C:\Users\Bill\Document	s\TestComplete 15 Projects\Notepad Autom	ation\Notepad Automation.pjs	
<u>File Edit View Test D</u> ebug <u>T</u> ools	<u>H</u> elp 👻		
: New - 🗁 🥁 🌇 - : 🖱 🤇	🖱 = i 🗣 🚳 🌄 = i 👧 = 🖞	🗸 🗾 Stop 🐞 🔋 Show Mobile	Screen 👻
Project Workspace Object Browser			
ितु Project Explorer ? 🗆 म 🗙	📝 Workspace		
🛛 🖓 🕞 🙆 🖓 🏠 🔧	👚 Start Page 🗴 🤨 Notepad1 🛛		
Y 😯 Project Suite 'Notepad Automation' (1 pr	Temporary Variables		
Notepad1 Motepad1 Morepad1	Add Variable 🔟 Delete		
	Name	Туре	Default Value

2. Add 'notepad.exe' as a Tested App

You may need to Add the 'TestedApps' item to your project. (right click on the project node, select 'New Item', then 'TestedApps'. Then create a new Tested App :

- a. select 'Generic Windows Application'
- b. for 'Application' just enter "notepad.exe"
- c. click the 'Finish' button

Project Workspace Object Browser		
Image: Comparison of the compar	Workspace	? 🗆 X
· 🕼 📮 🖳 🕰 🌾 -	🖌 Start Page × 🖸 Notepad1 × 🛅 TestedApps ×	•
Project Suite 'Notepad Automation' (1 pr V Project Suite 'Notepad1	notepad notepad.exe	Basic Parameters
Execution Plan		Name: notepad Run Mode: Simple 🔹
NameMapping		Count: 1 🗘
✓ Le Script +		Autorun application on recording
V TestedApps		✓ Launch application from tests
> Project Suite Logs		
		🥡 Run-Mode Parameters
		Application:
		notepad.exe 🔹 🗁
		Command-line parameters:
		Working folder:
		× 🖻

- 3. Record a keyword script where you...
 - a. open notepad
 - b. add some text
 - c. close note pad (click 'Don't save' when closing notepad)
 - d. discard / don't save

Then save this keyword script as 'Test2'.

Project Workspace Object Browser						
ित् Project Explorer ? 🗆 म 🗙	🗾 Workspace					? □ ×
: 🕰 📮 🕞 🖳 🕼 🏠 -	🕈 Start Page 🛛 🕑 Note	epad1 × 📔 Te	stedApps × 🔂 Test2 ×			*
Project Suite 'Notepad Automation' (1 pr	Operations #	👧 Record [Run 🚔 Add Paramet	ter 🔏 Add Variable 🕞	$\bigcirc \uparrow \downarrow \leftarrow \rightarrow$	
Execution Plan	Search P	Item	Ope	ration	Value	Description
✓ → KeywordTests +	Test Actions	Run T	estedApp noter	bad		Runs the "notepad" tested applicat
Test1	h On-Screen Action	v 🦲 notep	ad			
Test2	Run TestedApp	√ 🗆 w	ndNotepad			
NameMapping	Manu Aufers	2	Edit Click			Clicks the 'Edit' object.
✓ Cin Script +	Menu Action	2	Edit Keys		"some tes[BS]xt that we'll los[BS]ose"	Enters 'some tes[BS]xt that we'll lo
Unit1	No Process Action	🚬 🚽 🛌 w	ndNotepad Close			Closes the 'wndNotepad' window.
✓ TestedApps	💁 Run Keyword Test	🗸 🗌 d	gNotepad			
notepad	Run Script Routine		btnDontSave ClickE	Button		Clicks the 'btnDontSave' button.
> 🍖 Project Suite Logs	🕒 Run Test					
	Run Code Snippet					
	Call Object Method					
	Test Steps 🔀 Variables	Parameters				

4. Add this test to your Execution Plan

Project Workspace	Object Brows	er										
C Project Explore	r	?□ ╄ ×	🗾 Workspace								?	□ ×
🗳 📮 🖪	2	1 -	TestedApps ×	🔒 т	est2 * × 🔒 Keywor	rd Test Log [Test2] 07/02	2/2 × (Notepad	11 ×		4	⊦⇒
Y Project Suite	'Notepad Autor	mation' (1 pr	New Test Item	P	New Child Test Item	≩ ⊑ ≙ 🛱 🛍			$\downarrow \leftarrow \rightarrow$			
Exec	ution Plan		Name	Te	Execution entity	Parameters	Count	Timeou	On error	Link to external	Environ	«
🗸 🔂 Keyw	ordTests	+	🗸 🔂 Test2	\checkmark	KeywordTests - T	[none]	1 🗘	0 🛟	Use proj 🔻		0	
	est1 est2 Mapping											

5. Run this project several times locally. Make sure that it runs consistently and passes consistently.

Project Workspace Object Browser							
C Project Explorer ? 🗆 🕂 ×	Vorkspace	? □ ×					
: 🕰 📮 🖪 🙆 🙆 🔹 -	👬 Start Page 🗴 🔨 Notepad1 × 🛅 TestedApps × 🔂 Test2 × 🐕 Notepad1 19/02/2022 12:51:46 ×	-					
Project Suite 'Notepad Automation' (1 pr	🗄 🕞 Export Summary as JUnit 🎅 Export full log to 🛋 Share 🖂 Send via E- <u>M</u> ail 🖶 <u>P</u> rint						
Execution Plan	Test Execution Summary						
Constant Apping Constant Apple Constant Apple Constant Apple Constant Apple Constant Apple	↓ Test2 ↓ Notepad1 ↓ Signt ↓ Lighti ↓ Test Cases Run: 2 ↓ Failed: 0 ● Test Cases Run: 2 ● Failed: 0 ● Warnings: 0 ● Passed: 2 Execution duration: 00:00:12						
Project Suite Logs	General						
Image: Second Category (1992) Image: Second Category (1992) <th>Control executed: 2 Control Failed: 0 🕼 Warning: 0 Or Passed: 2 (100%)</th> <th></th>	Control executed: 2 Control Failed: 0 🕼 Warning: 0 Or Passed: 2 (100%)						
Notepad1 19/02/2022 12:51:46	Test Case Start Time 🔺	Duration					
Notepad 1 16/02/2022 00:52:40	♥ Test2 19/02/2022 12:51:46	00:00:11					
	log_everything 19/02/2022 12:51:58	00:00:01					

Don't get caught up in developing some advanced script or even something that looks exactly like the script above. You just need something that runs, and passes, consistently.

Stage 2 - Project Folder Share

Next, we need to make sure that this script can be shared across multiple machines. We need to share the project and the code between your local machine/laptop and your virtual machine. We'll do this using a windows file share.

Now you may already have a file share or you may need to set one up. If you need to set one up then follow these steps:

1. right click on your Project suite node and select 'Show in explorer'

TestComplete - C:	Rena <u>m</u> e		epad Automation\Notepad Au
File Edit View T	Remo <u>v</u> e		
: New - 🏳 🏳	Save	Ctrl+S	- 🔍 - Stop
	Save C <u>u</u> rrent As		
Project Workspace Ot	Show in CucumberStudio		
C Project Explorer	Show in Explorer		
: 🕰 📮 🔂 🧯 =	Co <u>l</u> lapse All	Ctrl+Alt+C	pad1 × 🛅 TestedApps ×
Project Suite 'No	E <u>x</u> pand All	Ctrl+Alt+A	it 🔂 Export full <u>l</u> og to
Execution Plan			

2. Now we know where our Project suite stored, we can share this folder.

- a. go up one level in explorer
- b. right click and select 'properties'
- c. on the 'Sharing' tab select the 'share' button

ts > TestComplet	te 15 Projects	Delete	cut	ipl
Name	^	Rename		т
Notepad A	utomation	Properties		F
System Vol	ume Informatio	on	08/02/2022 07:46	F
💿 RunMyTest	s		08/02/2022 08:36	۷
🍓 TestExecut	e_T68891162121	12SLM	07/02/2022 12:44	A

5 Projects

liew		📕 Note	pad Auto	mation P	roperties		×
nts	> TestComplete 15 Project	General	Sharing	Security	Previous Versions	Customise	
	Name	Netwo	Notepa	l Folder Sh ad Automa	haring tion		
	Notepad Automation		Shared	3			
	System Volume Informa	Netwo	ork Path:				
	RunMyTests	1.00	1000		and the Designation	Card Card In	
	🎲 TestExecute_T68891162	S	hare				
		Advar	nced Sharir	ng			
		Set c	ustom pem	nissions, cr	reate multiple shares	and set othe	r

3. Once you've shared your folder you will want to copy the full UNC path so that you can access the share from your VM.

2		×	
←	a Network access		
	Your folder is shared. You can <u>email</u> someone links to these shared items or <u>copy</u> and paste the links into another app.		
	Individual Items		
	Notepad Automation		
			×
	<u>Shared items aren't accessible when your computer is asleep.</u> <u>Show me all of the network shares on this computer.</u>		

You'll have a UNC path something like this...

//YOUR_MACHINE_NAME/Users/USER_NAME/Documents/TestComplete%2014%20Proje cts/NotepadAutomation

Keep a record of this as we'll need it on your VM in a minute

Stage 3 - Virtual Machine setup

We need a machine to 'develop' our tests on. We also need a separate machine to 'execute' our tests on. You can use your desktop or laptop for development with TestComplete. Then use a Virtual Machine, running TestExecute, to execute your test. This gives you the capability for testing/shake down of your tests or for formal execution runs. Either way you free up your main machine and increase your productivity.

If you already have a Virtual Machine, you can skip through to Stage 4 below. If you don't already have a VM then you can download Virtual Box from here and run the installer:

https://www.virtualbox.org/wiki/Downloads

From the same web page, download and Install the Virtual Box Extension Pack (which includes support for Remote Desktop connections).

```
VirtualBox 6.1.32 Oracle VM VirtualBox Extension Pack
```

```
    ➡All supported platforms
```

Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP, disk encryption, NVMe and PXE boot for Intel cards. See this chapter from the User Manual for an introduction to this Extension Pack. The Extension Pack binaries are released under the VirtualBox Personal Use and Evaluation License (PUEL). Please install the same version extension pack as your installed version of VirtualBox.

Once you're running with Virtual Box then you can follow these steps to create and configure a Virtual Machine.

- 1. Download and extract a Virtual Box Windows image from: https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/
- 2. Double click the .ova (Open Virtualization Archive) file and import this file in Virtual Box

🗿 Oracle VM VirtualBox Manager		_	
Tools	Appliance settings These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.	<u> </u>	
	Sound Card ICH AC97 Network Adapter Intel PRO/1000 MT Desktop (82540EM) Storage Controller (IDE) PIIX4 Storage Controller (IDE) PIIX4 Virtual Disk Image IE11 - Win81-disk001.vmdk Base Folder C:\Users\Bill\VirtualBox VMs Primary Group /		
	Machine Base Folder: C:\Users\Bill\VirtualBox VMs MAC Address Policy: Include only NAT network adapter MAC addresses Additional Options: Import hard drives as VDI Appliance is not signed Restore Defaults Import Cancel		

3. Start the new box



4. Check this new box runs okay, capture the IP address, User Name and password (if you've downloaded the Microsoft image then the user name will be "IEUser" and the password "PasswOrd1")

5. Enable "Remote Access" on this machine

🕌 IE11 - Win	81 [Running] - Oracle VM VirtualBox	
File Machine	e View Input Devices Help	
	System Properties	
	System Properties × Computer Name Hardware Advanced System Protection Remote Remote Image: Advanced System Protection Remote Image: Advanced System Protection Remote Image: Advanced Image: Advanced Advanced Image: Advanced Image: Advanced Advanced Remote Desktop Image: Advanced Image: Advanced	vnloaded archive) b y after the OS trial o
	Choose an option, and then specify who can connect. Don't allow remote connections to this computer Allow remote connections to this computer Allow connections only from computers running Remote Desktop with Network Level Authentication (recommended) Help me choose Select Users OK Cancel Apply	eriod: on versions of Micro l license on the desi s, you need to com ll be done automati ministrative commar rist boot. t boot. You will see left (in the system tr

6. Close and Power Off the new box



7. Configure an optical drive and add VBoxGuestAdditions.iso

	11 - Win81 - Settin	igs		?	×	×
	General	Storage				
	System	Storage Devices	💹 IE11 - Win81 - Optical Disk Selector			
	Display	Controller	Medium 🔓			
9	Storage					
	Audio		Add Create Rerresh			
	Network		Name Virtual Size V Not Attached			
	Serial Ports		ad-hoc.viso 0 B VBoxGuestAdditions.iso 58.37 MB			
Ď	USB		3			
	Shared Folders					
	User Interface					
			<			
			Search By Name 🔻			_
			<u></u>		Ch	100
						T
		Invalid se	ttings detected 👖 🥌 OK	Can	cel	T

🗿 Oracle VM Virtua	alBox Manager	—		\times
File Mac 😳 IE11	1 - Win81 - Settin	32	?	\times
П тс	General	Network		
	System	Adapter 1 Adapter 2 Adapter 3 Adapter 4		
	Display	C Enable Network Adapter		
	Storage	Attached to: Bridged Adapter		_
· •	Audio	Name: Intel(R) Dual Band Wireless-AC 3165 Advanced		•
	Network			
	Serial Ports			
🖉 (USB			
؛ 🗖 ا	Shared Folders			
	User Interface			
2				
н		Invalid settings detected 🥂 OK	Can	cel

8. Configure the network to be a 'Bridged Adapter"

9. Restart the box, go to file explorer, open VirtualBox Guest Additions and run through the install

IE Versi		
OS Ver	😵 Oracle VM VirtualBox Guest Additions 6.1.32 Setup 🛛 🗖 🔼 😽	
Service		
User Na	Wolcomo to Oraclo VM VirtualBox	
Passwo		N
	Guest Additions 6.1.32 Setup	√
Snans	Colore will write your through the installation of Oracle 104	
📑 I 🛃 🔝 🖬 I	VirtualBox Guest Additions 6, 1, 32.	
File Computer	inclue.	v 😮
	It is recommended that you close all other applications	
🔄 🕣 🕤 🛧 🖪	PC	Q
	computer.	
🔆 Favorites	Click Next to continue	
Desktop	Click Next to containde.	
Downloads		
B D to d		
🕍 Recent places		
🖳 This PC		
S Network	Next > Cancel	
TILLWOIK		
	Devices and drives (2)	
	Windows 81 (C:)	
	Additions	
	18.8 GB free of 39.9 GB 0 bytes free of 58.3 MB	

- 7. Shut the machine down, then...
- 8. Restart the new box

At this point you should have a Virtual Machine available and running. You must be able to RDP on to this machine and create a desktop session. From here we'll need to install TestExecute.

- 9. RDP onto the Virtual Machine use 127.0.0.1:5958 (you must RDP onto the "host" machine on the specified port) (more info here if you have problems: <u>https://marcowuen.wordpress.com/2016/10/31/troubleshooting-virtualbox-vrde-remote-desktop-connections/</u>)
- 10. With the shared folder you created on Desktop/Laptop machine, use the UNC recorded (above in Stage 2) and paste the address into the File Explorer address bar (you will probably have to change the '' separators to '/' seperators to get this to work)

\\YOUR_MACHINE_NAME\Users\USER_NAME\Documents\TestComplete 14 Projects\NotepadAutomation

You should be able to access this shared directory on your Virtual Machine something like this...



At this point, File Explorer on your VM should show the mounted driver that contains your TestComplete projects



11. Install TestExecute on this Virtual Machine You can download a TestExectue trial using this link: <u>https://smartbear.com/services/trials/downloadtrial.aspx?file=TestExecute15</u> <u>30.exe</u>

Stage 4 - Run the TestExecute Project on the Virtual Machine

We should now be able to run our project, written in TestComplete, on our VM using TestExecute. When running a project from a shared drive like this, it's known as running projects in 'Shared Mode'.

1. Start TestExecute and open the 'notepad1.mds' project file





When you open this project from a shared drive you'll be presented with this prompt...

TestExecute	
The project suite 'G:\Notepad Automation\Notepad Automation.pjs' is located on a network drive. Do you want to work with it in shared mode?	
YesNoHelp \checkmark Remember the answer for this project suite and do not ask again.	

You'll want to select 'Yes' and then save the configuration file in the same directory as your project suite. You should see a directory created like this...

Choose a folder for the local configuration file							
€ ∋ - ↑ 🎚	(e) → ↑ (i) ≪ Test → Notepad Automation → ↓ ♥ Search Notepad Automation 𝒫						
Organize 🔻 New	folder		∷ • 🔞				
Desktop	^ Name	Date modified	Туре				
Downloads	📃 🍌 Log	2/7/2022 4:48 AM	File folder				
🕍 Recent places	📕 Notepad1	2/7/2022 4:49 AM	File folder				
This DC	💠 Notepad Automation.pjs	2/7/2022 4:48 AM	TestExecute proj				
I Inis PC	Notepad Automation.pjs.tcLS	2/7/2022 4:48 AM	TCLS File				
🗣 Network	Notepad Automation.tcCfgExtender	2/7/2022 4:50 AM	TCCFGEXTENDE				
	v <		>				
File name:	[E11WIN8_1]		¥				
Save as type:			~				
) Hide Folders		Save	Cancel .:i				

In shared mode, TestComplete creates separate configuration files for each team member who runs the tests. In addition, TestComplete organizes the test logs by creating separate log folders in the log tree for each test computer where the tests run.

You'll learn more about this here:

https://support.smartbear.com/testcomplete/docs/working-with/teamwork/sharing-projects.html

Finally run your project on your from your VM. You can this by either...

a. clicking on the TestExecute icon in the Task Bar and selecting 'Run'

or

b. pressing 'shift' and "F3"

After the run has completed TestExecute may try to open the log file in your default browser. This may be a .mht file which is only supported in IE. If you have IE on your VM you should see something like this...

	er\AppData\Local\Temp\tcRes 🔎 = 🖒	TestComplete 15 Log ×	- □ × + ★ #
File Edit View Favorites	Tools Help		
Test Execution Sur	mmary		
No	tepad Automation • Test Cases Run: 1 • Failed:	0 ● Warnings: 0 ● Passe	d: 1
Exe	ecution duration: 00:00:10		
\pm General			
C Total executed: 1	S Failed: 0 🛦 Warning: 0	✓ Passed: 1 (100%)	
Test Case		Start Time 🔺	Duration
Test2		2/8/2022 12:16:57 AM	00:00:10

If you don't have IE on your machine you can click on the TestExecute icon in the Task Bar and select 'Export Result'

		Open	i
	<u> </u>	Run	Shift+F3
<u>e</u>		Stop	Shift+F2
	15	View Result	•
		Export Result	•
	····•	Export Summary as JUnit	+
	7	Create Jira Issue	•
	eula	Post Defect to Bugzilla	
		Post Defect to QAComplete	· · · ·
		Issue Tracking	•
		Run Mobile Troubleshooter	η
		Options	
		Install Extensions	
		Install Script Extensions	
		Help	
		Check for Updates	
		Contact Support Team	
		About	
		Switch to Key-Based Licensing	
		Log out Bill Echlin	
		Manage License	
	1	Exit	

We'll look at creating logs and distributing logs in more detail in a bit. For now though, we should be in a position where...

- 1. you can develop tests on your local desktop/laptop
- 2. you can execute tests on your virtual machine

In the next section we'll look at how we can trigger these tests from our local Desktop/Laptop machine and then let them run remotely on our virtual machine. We'll be able to develop on one machine, run and test your tests on another machine.

Stage 5 – Remotely Triggering Test Execution

There are a number of ways to trigger remote execution. You could use tools like PsExec.exe or Jenkins. The simplest way though is to use the inbuilt Test Runner REST API that comes with TestComplete and TestExecute. We'll set things up so that we can make a simple command line REST API request from our local Desktop/Laptop machine, to our VM where we'll start the test execution.

It's recommended that you only use this on internal networks. Opening up Ports that allow execution in this manner can compromise your computers security. With our VM setup running on Virtual Box we'll be okay but you do need to be aware of the important information set out here...

https://support.smartbear.com/testcomplete/docs/reference/apis/test-runner.html



First on the VM, from the task bar close TestExecute by clicking on "Exit".

Then again on the VM start 'Task Manager' and make sure that TestCompleteService15.exe is running. This should have been installed and started when you installed TestExecute.

P	Task Manager						:
File Options View							
Processes Performance App history S	Startup U	sers Det	ails Services				
*			3%	33%	0%	0%	
Name	Name Status			Memory	Disk	Network	
Image: Spooler SubSystem App	0%	2.1 MB	0 MB/s	0 Mbps	^		
sshd.exe			0%	1.1 MB	0 MB/s	0 Mbps	
▷ 📑 TestCompleteService15.exe (32			0%	2.5 MB	0 MB/s	0 Mbps	
VirtualBox Guest Additions Servi			0%	1.8 MB	0 MB/s	0 Mbps	

Next on the VM we need to run this batch file that will enable and turn on the api.

1. Open an explorer window and navigate to this directory: C:\Program Files (x86)\SmartBear\TestExecute 15\Bin 2. Run (right click) the "TestCompleteService15RestApiEnable.cmd" command as an administrator



3. Type "Y" at the prompt and accept the default port of 1880



4. Change the firewall settings on the VM to allow the Api calls

@	Windows Firewall with Advanced Se	- • ×	
File Action View Help Image: Constraint of the second seco			
Windows Firewall with Advance Inbound Rules Outbound Rules Connection Security Rules	Inbound Rules Name OpenSSH OpenSSH OpenSSH Ssh SSHD SSHD	Gro ^	Actions Inbound Rules New Rule Filter by Profil 2 Filter by State Filter by Group View

5. click on Inbound rules and add a New Rule with the settings below:

Rule Type: Port Protocol and Ports: TCP and specific port 1880 Action: Allow the connection Profile: Domain & Private & Public Name: TestComplete REST

6. Allow interactive sessions by configuring the following group policies

Run "gpedit.msc" Local Computer Policy > Computer Configuration > Administrative Templates > All Settings Disable the following policies:

- Always prompt client for password upon connection
- Prompt for credentials on the client computer

Once this is complete, from your Local Desktop/Laptop machine you should be able to access the Swagger service definition from your browser. We'll test this out next...

7. On your Local machine open a browser like Chrome and navigate to this url (you may find you have to do this with an IP address in place of the host name we've provided in this example).

http://ie11win8_1:1880/1/docs/index.html (you may need to change your host name here if it's not the default ie11win8_1)

Then Login with the user name and password below User: IEUser Password: PasswOrd!

G i=11win8 1-1880/1/docs/index.ht X +	
← → C ③ ie11win8_1:1880/1/docs/index.html	
Sign in http://ie11win8_1: Your connection to Username IEUs Password	1:1880 to this site is not private User Sign in Cancel

From here we can test our REST Api configuration by registering our existing Project Suite and making sure we can start TestExecute.

8. Register the project by going to this URL for the swagger UI that we use to register the Project Suite

http://ie11win8_1:1880/1/docs/index.html#/default/putProjectSuite (you may need to change your host name here if it's not the default ie11win8_1)

9. Enter the following URI value for the project suites location on the VMs file system

file:///G:/Notepad Automation/Notepad Automation.pjs

10. Click execute and check you get back the suite id. We'll need this suite id to actually trigger the test



Now we know that the remote execution connection is working we can create a quick powershell script to run our tests on demand. Copy the script (below in the appendix) to create a powershell script that will trigger your test runs on the remote machine.

- 11. Open notepad, copy in the script and save the file as "runMyTests.ps1"
- 12. Edit and update the \$vm_host_name and \$vm_host_port values if they differ from the default
- 13. You may need to set your powershell execution policy by executing the following command in a powershell session

Set-ExecutionPolicy -ExecutionPolicy Unrestricted

14. Run the script and check the output



15. You should see the script go through the process of...

i. registering the project suite
ii. starting TestExecute and registering the instance
iii. checking the status prior to execution
iv. running your script
v. checking status as it waits for completion
vi. getting the execution summary reponse
vii. closing the instance of TestExecute

All of this should result in the VM connection logging out whilst the test run executes in a separate windows session.

All we need to do now is sort out the logging so that we can capture the test result log on our local machine.

Stage 6 - Logging the Test Results

At this point we have a VM setup. We've created a file share which means we can develop tests on one machine and run them on the VM. We've configured the REST Api so that we have the ability to trigger executions on the VM remotely. We've developed a PowerShell script that allows us to trigger the test run on the VM with the simple execution of this script.

All we need now is to find a way to log the test results in a useful format and pull them back to the local machine where we can see them. We can do that by enhancing our original TestComplete project and making sure the log file is saved from the VM to the file share we've created.

1. Working on your local Desktop/Laptop with TestComplete, edit Unit1 in the Script section of your project



2. In your empty script unit, 'Unit1' add the following python code

```
def log_everything():
    Now1 = aqDateTime.Now()
    Now1string = aqConvert.DateTimeToFormatStr(Now1,
    "%Y_%m_%d_%H_%M")
    Log.Message("Log folder is : " + Now1string)
    Log.SaveResultsAs("G:\\Notepad Automation\\Notepad1_logs\\"
+ Now1string, lsHTML, True, 1)
```

We're assuming here that your project is set to 'Python'. If you created your project with a different language you may need to adjust this code slightly.

3. In your script you may have to adjust the path in the Log.SaveResultAs method to your specific shared directory. If you've followed everything as listed so far then you should be okay with this directory. At this point your script should look something like this in TestComplete.



4. On your Execution plan add your new log_everything() method.

Project Suite 'Notepad Automation' (1 pr	: 📭 New Test Item	New C	hild Test Item 📭 🚘 😭	m 🖪 🛛	B 1	$\downarrow \leftarrow$	
Vierad1	Name	Tes	Execution entity	Parameters		Count	Γ
V 🔂 KeywordTests +	Test2	✓	KeywordTests - Test2 ····	[none]		1 🗘	
Test1	🗸 🗸 log_everything	\checkmark	Script\Unit1 - log_everything	[none]		1	
Test2							

5. Save your project.

At this point you're saving on the file share and the changes made will be available to TestExecute on your VM immediately. All we need to do to check this works is run our runMyTest.ps1 PowerShell script

6. On your desktop, or wherever your runMyTest.ps1 script is located, run your PowerShell script



7. Check that the script runs successfully



8. On your local machine, your file share, should have a new logs directory 'Notepad1_logs'



9. In this folder find the sub directory with the date / time file name and open the folder

Notepad1_	logs > 2022_02_19_07_32	✓ ひ Search 2022	2_
* ^	Name	Date modified	
*	ScriptTestLogUnit1log_everythi	19/02/2022 15:32	
*	Test2KeywordTestsTest2	19/02/2022 15:32	
*	🌋 _ProjectLog	19/02/2022 15:32	
*	🌋 _root	19/02/2022 15:32	
	📓 aqds	19/02/2022 15:32	
	🌋 caption	19/02/2022 15:32	
	👼 default	19/02/2022 15:32	
	icons	19/02/2022 15:32	
ting-vic	💿 index 🛛 👞	19/02/2022 15:32	

10. In this folder you'll find the index.html file which you can open with your browser. You should see a results page like this...



Your test runs on your VM are now easily accessible on your local Desktop/Laptop machine

This last stage demonstrates perfectly how you can update your automation scripts on your local machine, run the scripts on your VM and then see the results in a browser on your local machine. You now have a very efficient way to develop, test and run your automated tests.

Appendix A – PowerShell Script To Execute Tests Remotely

a = 1

```
# VM Host name
$vm_host_name = "iellwin8_1"
$vm host port = "1880"
# Register the TestComplete project suite
$suiteid response = cmd /c curl -s -X "PUT"
"http://${vm_host_name}:${vm_host_port}/1/suites?uri=file%3A%2F%2F%2F%3A%2FNotepad
%20Automation%2FNotepad%20Automation.pjs" -H "accept: application/json" -u
"IEUser:Passw0rd!"
$suiteid = ($suiteid response | Out-String | ConvertFrom-Json) | Where-Object
result | ForEach-Object {"$($_.result.suite)"}
echo "Suite ID is $suiteid"
$instanceid response = cmd /c curl -s -X "POST"
"http://${vm host name}:${vm host port}/1/suites/$suiteid/create instance?screen wi
dth=1280%26screen_height=1024%26color_depth=32" -H "accept: application/json" -u
"IEUser:Passw0rd!"
$instanceid = ($instanceid_response | Out-String | ConvertFrom-Json) | Where-Object
result | ForEach-Object {"$($_.result.instance)"}
echo "Instance ID is $instanceid"
$instancestatus response = cmd /c curl -s -X "GET"
"http://${vm_host_name}:${vm_host_port}/1/instances/$instanceid" -H "accept:
application/json"-u "IEUser:Passw0rd!"
echo "Instance status response is ... $instancestatus response"
$instancestatus = ($instancestatus response | Out-String | ConvertFrom-Json) |
Where-Object result | ForEach-Object {"$($ .result.state)"}
echo "Instance status is $instancestatus"
```

```
Do
{
echo "Waiting 10 seconds"
Start-Sleep -Second 10
$instancestatus response = cmd /c curl -s -X "GET"
"http://${vm host name}:${vm host port}/1/instances/$instanceid" -H "accept:
application/json"-u "IEUser:Passw0rd!"
echo "Instance status response is ... $instancestatus_response"
$instancestatus = ($instancestatus_response | Out-String | ConvertFrom-Json) |
Where-Object result | ForEach-Object {"$($_.result.state)"}
echo "Instance status is $instancestatus"
$a++
if ($a -eq 7)
{
echo "Giving up"
Exit
}
} While ($instancestatus -ne "ready")
$run response = cmd /c curl -s -X "POST"
"http://${vm_host_name}:${vm_host_port}/1/instances/$instanceid/run?project=Notepad
1" -H "accept: application/json" -u "IEUser:Passw0rd!"
echo "Run response is $run_response"
$run status = ($run response | Out-String | ConvertFrom-Json) | Where-Object result
| ForEach-Object {"$($ .success)"}
echo "Run status is $run status"
$instancestatus = ($instancestatus response | Out-String | ConvertFrom-Json) |
Where-Object result | ForEach-Object {"$($_.result.state)"}
echo "Instance status is $instancestatus"
$a = 1
Do
echo "Waiting 10 seconds"
Start-Sleep -Second 10
$instancestatus response = cmd /c curl -s -X "GET"
"http://${vm host name}:${vm host port}/1/instances/$instanceid" -H "accept:
application/json" -u "IEUser:Passw0rd!"
echo "Instance status response is ... $instancestatus response"
$instancestatus = ($instancestatus_response | Out-String | ConvertFrom-Json) |
Where-Object result | ForEach-Object {"$($_.result.state)"}
echo "Instance status is $instancestatus"
$a++
if ($a -eq 7)
echo "Giving up"
Exit
}
} While ($instancestatus -ne "ready")
$runsummary response = cmd /c curl -s -X "GET"
"http://${vm host name}:${vm host port}/1/instances/$instanceid/summary" -H
"accept: application/json" -u "IEUser:PasswOrd!"
echo "Run summary response is ... $runsummary response"
$closeisntance response = cmd /c curl -s -X "DELETE"
"http://${vm_host_name}:${vm_host_port}/1/instances/$instanceid" -H "accept:
application/json"-u "IEUser:Passw0rd!"
echo "Close instance response is .... $closeisntance_response "
```