# Appendix C: Example from Blog – Part 2: Generating Documentation with the Script

To recap Part 1 – We basically walked through the example runbook used in this guide. Now that we have a basic understanding of what our Example Runbook does (*Creates a folder, creates a file, writes data to file and also logs event log messages*), we will now walk through the process of prepping this runbook to be documented with the script.

**Export the Runbook - then Create the Images**

What I mean by that is that if you have been a good little Sysadmin - and have done the recommended reading I recommended in documentation for my script:

* [TechNet Blog - Automation–System Center 2012 – Orchestrator Best Practice Series–Naming Conventions](http://blogs.technet.com/b/privatecloud/archive/2013/03/08/automation-system-center-2012-orchestrator-best-practice-series-naming-conventions.aspx)
* [System Center Orchestrator 2012 – Best Practices – Runbook Design](https://scorch.codeplex.com/releases/view/119073)

This is due to the fact that all of that information (*i.e. things with system generated GUIDs – GUIDs that don’t exist anywhere other than your export source’s* ***Orchestrator Database***) needs to be “**scrubbed**” from the export (*which is not one of those things you just love to do each time you export a runbook…*) if you ever plan to import that runbook into another environment that is not the EXACT same environment you exported it from in the first place!

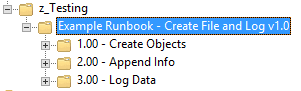
For our purposes though, **we will check every option**. Unlike an export used for transfer to different environments, this export’s **sole purpose** is to provide enough information as possible (*i.e. least calls to the database for the script necessary*) so we can generate some useful reports that we can refer others to later **without them needing Runbook Designer access** *(initially)* **or expertise**.

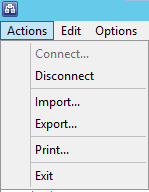
A special item to note at this point is that – the runbook I included for an example does not have any connection to any variables or globally scoped environment variables – so I have not included that information in the example runbook.

If I HAD included global variables (*like connection information, globally accessible server names, etc.)* then they would have had their GUIDs updated to be a little more understandable to the reader as well.

**Export the Runbook**

So first, select the root folder (where the master runbook resides) where you want to capture all the runbooks contained within the **sub-folders of the root document:**

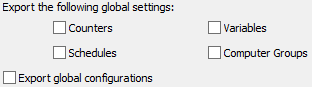


Then, from the **Runbook Designer** menu, select “**Actions**” > “**Export**”:  


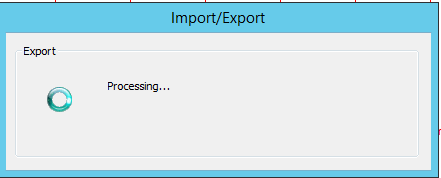
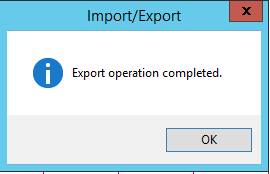
For the **File Location,** save the file as whatever you want:



**Check** the “**Export the runbooks in sub folders**” option:  


  
**Example *(Above)*:** Uncheck all options **except Variables** (*if used within the runbooks – ours does not)*

When you have all your options configured, click the **Finish** button to complete the export.

Now that we have an export, we need to get our “**Snipping**” hat on and get our Runbook images

**Create the Activity Icons**

Next, we want to make sure our documentation contains no broken or confusing images, so we need to create our “**activity icons**”.

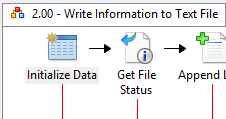
The documentation supplied with the source code contains the steps involved with creating additional icons. For your reference (*so you don’t think I’m leaving you high and dry*) – I have a “starter set” already going. Here’s a quick breakdown of Activity Icons already included with the script:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Icon | ObjectTypeName | Index # | Activity Icon File Name | Integration Pack |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_0_db_query.png | **Query Database** | 0 | ico\_0\_db\_query.png | Utilities |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_1_dotnet_script.png | **Run .Net Script** | 1 | ico\_1\_dotnet\_script.png | System |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_2_event_log.png | **Send Event Log Message** | 2 | ico\_2\_event\_log.png | Notification |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_3_initialize.png | **Custom Start** | 3 | ico\_3\_initialize.png | Runbook Control |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_4_return_data.png | **Publish Policy Data** | 4 | ico\_4\_return\_data.png | Runbook Control |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_5_unknown_activitytype.png | **Default (Unknown) Activity** | 5 | Ico\_5\_unknown\_activitytype.png | All Activities you don’t have custom icons for will use this as the default |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_6_compare_values.png | **Compare Values** | 6 | ico\_6\_compare\_values.png | Utilities |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_7_ad_createuser.png | **Create User** | 7 | ico\_7\_ad\_createuser.png | Active Directory Integration Pack |
| ~~F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_8_create_folder.png~~ | **Create Folder** | 8 | ico\_8\_create\_folder.png | File Management |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_9_exch_create_mailbox.png | **Enable Mailbox** | 9 | ico\_9\_exch\_create\_mailbox.png | Exchange Admin Integration Pack |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_10_ad_enable_user.png | **Enable User** | 10 | ico\_10\_ad\_enable\_user.png | Active Directory Integration Pack |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_11_junction.png | **Junction** | 11 | ico\_11\_junction.png | Runbook Control |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_12_get_file_status.png | **Get File Status** | 12 | Ico\_12\_get\_file\_status.png | File Management |
| F:\Users\Michael\Desktop\SCORCHTESTEXPORTSCRIPTS\icons\activities\ico_13_append_line.png | **Append Line** | 13 | Ico\_13\_append\_line.png | Text File Management |
|  | **Invoke Runbook** | 14 | Ico\_14\_invoke\_runbook.png | Runbook Control |
| 🡪 | **Link** | NA | image\_link\_activity.png | Used to represent the actual link between two objects |

**Create the Runbook “Map” Images**

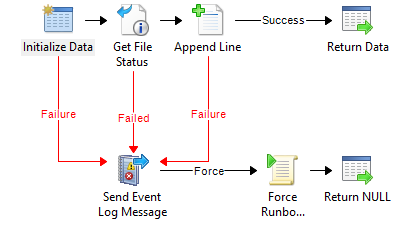
The last thing we need to do before we run the script is create the images that we will use to create an interactive runbook “Map” that will use some CSS and JavaScript to provide jump-points to other activity reports for a certain runbook.

**These steps are also outlined in my documentation for the script – but I will be extra clear here:**

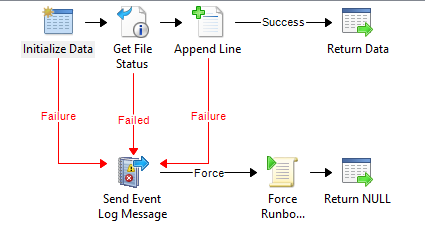
* Snip the image from the upper most left corner of the runbook image, **but do NOT** include the “black border” or the “runbook tab” when taking the snip. i.e.:  
    
  

Start the snip in this corner where the tab bottom and left border intersect

**This is good:**



**This, not so good**:



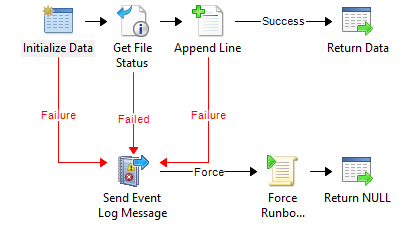
Notice how the border got included with this snip – any amount of “offset” **may cause issues with your runbook maps**

Name your images exactly as the runbooks are named, but by removing spaces.

For example, the runbook image for “**2.00 – Write Information to Text File**” would be named:

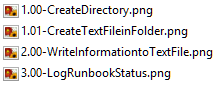


“**2.00-WriteInformationtoTextFile.png**”





The resulting images for each runbook we will be reporting on in the example we are using will look like so:



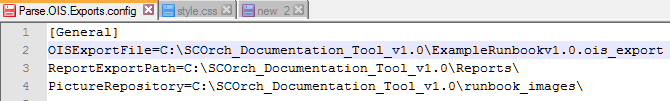
Once we have our images in place and our export is ready, we are finally ready to run the script!

**Tip:** If you’re images end up with some hot spots way off the mark, then make sure that the runbook image you snipped was actually “**Checked In**” in the Runbook Designer **before you export the runbooks**. I did that a couple times and spent a while digging through code before realizing the code wasn’t wrong, it was the fact that I had moved the activities x,y positions in the Runbook Designer after taking the snips and then had checked in the runbooks.

I should have checked in the Runbooks (*ALL of them you have images for!*) prior to taking the Snips, then everything would have been fine…

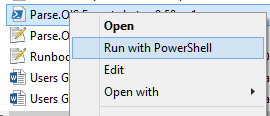
**Finally – Running the Script!**

First off, note the location of the runbook export you want to use, and open up the “**Parse.OIS.Exports.config**” file in your favorite text editor (I use Notepad++ most of the time). Edit the [Genera] > OISExportFile value with the path to your runbook OIS\_Export file:

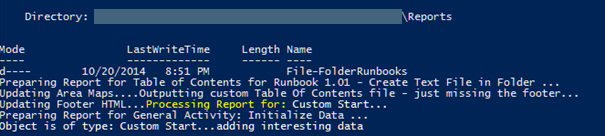


Ensure your database authentication (Integrated Auth) has been set, and your paths to the rest of the variables match the root directory of your script.

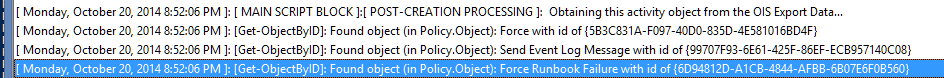
Once everything is in place, just right click the script and run it!



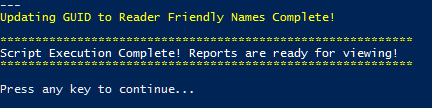
Now, just wait…



By using **CMTrace.exe** or **Trace32.exe** log viewing utilities, you can also open up the log file and watch the more verbose info:



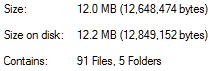
**When the script is complete** you will see the **“Script Execution Complete!”** message and the PoSH Command Prompt asking you to press a key to continue (*this will close the PowerShell console window*):



Looks like we generated 91 pages of documentation (interactive docs no less) in a little over one minute!!! Not Bad!!!

**(Start Time)** 

**(Finish Time)** 



**Now go to your reports and start checking them out!!!!**

**Please, please read the documentation!**

I spent a good amount of time trying to document every component I think could be that wasn’t the script itself (*aside from the “Creating your own Activity Icons” procedure*). I wanted to answer every question I could think of and cite any sources or links I used during my process.

These references, explanations, and links should all be in the user documentation available in my github repository.

If you still can’t find the answers to your questions after reading the supplied documentation, then by all means drop a comment on the site and I will try my best to give a useful answer!! ;)