Q&A Videos: Execute an Exe in Azure

|  |  |
| --- | --- |
| **Writer:** Frank Boucher | **Producer:** |
| **Audience:** .NET developers |  |
| **Video for** Executing an Exe in Azure | |
| **Type of video:** Q&A | |
| **Tips and tricks:**   * **Remember your script will be read aloud. Keep things as short and succinct as possible.** * **~150 words translates to roughly 1 minute of video time. Keep this in mind while drafting your script.** * **When at all possible, show don’t tell.** * **Stay focused on the main concept or problem you want to solve.** | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Scene** | **Narrator says** | **Customer sees** | **Describe the action on screen (add repro steps, if needed)** |
| **1** | To take advantage of existing code, you can call an executable or library DLL from Azure Functions. |  |  |
| **2** | Our sample app takes a parameter then returns a greeting with the local time. |  |  |
| **3** | Start by creating an Azure function in Visual Studio and copying your executable inside the folder structure. To always copy the files on deployments, go to the property panel and set the value to “Copy Always. |  | * Show where the file is located into the solution explorer (right panel) * Show the “Copy the Output Directory” property (right panel, bottom of the screen) and its value. |
| **4** | To use the executable’s outputs from inside the Azure Function, you need to set the properties to redirect the standard output.  You can then add any arguments the executable needs. Here the function parameter ‘name’ is passed to the executable and the output is then added to an http message to be returned by the Azure Function. |  | Looking at the code section.  On line 37 à  UseShellExecute = false  On line 38 à  RedirectStandardOutput = true |
| **5** | Let’s deploy and test it. Once deployed, in the Azure portal, find your function and run it from the Code + Text blade. |  | Read “Code + Text” as “Code plus Text” |
| **6** | It should return the same message as the executable. |  |  |
| **7** | To get the path used to reference the executable, go to the Advanced Tools in the Azure Portal and open the Kudu interface. |  |  |
| **8** | In this case you can see the path matches what we had in the code. |  | Last line in the terminal shows the path "home\site\wwwroot" |
|  |  |  | On line 31 à  The path used in the code "home\site\wwwroot"  Show side by side and highlight |
| **Final** | By calling your executables or DLLs from within Azure Functions, you can reuse existing code without having to recompile it. |  |  |

## Appendix

All Images and videos (face, monitor, and mixed) are available in SharePoint here: <https://microsoft.sharepoint.com/:f:/t/QnAVideos/EtDu8klEr5RPjPhbthDoqU0BHmQDjCVFMld9JJkVLkNUwQ?e=eB0QBv>

# Notes

* Scene 1: no still
* Scene 2: 00:11 - 00:17 (still 00:15)
* Scene 3: 00:19 - 00:52 (still 00:23 or 00:32)
* Scene 4: 00:53 - 01:35 (still 01:11 or 01:27)
* Scene 5: 01:37 - 01:47 (still 01:45)
* Scene 6: 01:48 - 01:57 (still 01:48 or 01:50)
* Scene 7: 01:58 - 02:06 (still 2:05)
* Scene 8: 02:09 - 02:43 (still 2:33 or 02:34)
* Scene 8: re-use still from 00:23 or 00:56

When in the Azure Portal can see the “Preview” logo in the top right corner. I don’t think it is a huge problem. However, if we can mask it, it will be better.