Windows based languages for job implementation with COM components

- Scope
- Language Overview
- Examples
    - COM component for example jobs
    - VBScript jobs calling COM components
        - VBScript job for Master
        - VBScript Job for Agent
    - PowerShell jobs calling COM components
        - PowerShell Job for Agent

Scope

- JobScheduler is evolving towards an architecture that allows more flexible use with Agents.
- Find an overview of supported languages for jobs that make use of COM components.

Language Overview

Master		Agent		
Language	**Architecture**	**Language**	**Architecture**	**Comments**
`VBScript`	JVM 64bit + Master 64bit • call 64bit components, e.g. COM • access to 64bit registry	`ScriptControl: VBScript`	No support for JVM 64bit + Agent	• Agent can be used with JVM 32bit and 64bit • ScriptControl is available for 32bit only • COM objects can be instantiated from 32bit architectures • Users of VBScript jobs on a Master 64bit should migrate to Power Shell Jobs for Agents
	JVM 32bit + Master 32bit • call 32bit components, e.g. COM • access to 32bit registry		JVM 32bit + Agent • call 32bit components, e.g. COM • access to 32bit registry	• ScriptControl works as a compatibility mode for VBScript Jobs • Minor syntactical changes to job scripts required
		`PowerShell`	JVM 64bit + Agent • call 64bit components, e.g. COM • access to 64bit registry	• COM objects can be instantiated from both 32bit and 64bit architectures • The architecture of the JVM in use determines if 32bit or 64bit COM components are referenced
			JVM 32bit + Agent • call 32bit components, e.g. COM • access to 32bit registry	
		`dotnet` (.NET)	same architectures as PowerShell	• Jobs can be implemented in any .NET language by use of the Job Implementation Interface • FEATURE AVAILABILITY STARTING FROM RELEASE 1.10.7

Examples

- Examples are available for download from scripting.zip
- Unzip the archive in the `./config/live` folder of your JobScheduler Master, a sub-directory `scripting` will be created for job-related objects.

COM component for example jobs

- From the attached archive scripting.zip register the COM component `ComComponent.dll` that implements a sample class. The .dll has been compiled using the *AnyCPU* model and has to be registered by the user.
- Register the COM component for your respective architecture. The path to the .NET Framework might be different for your environment:
  - 32bit
    - `C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe ComComponent.dll /codebase /tlb /nologo`
  - 64bit
    - `C:\Windows\Microsoft.NET\Framework64\v4.0.30319\regasm.exe ComComponent.dll /codebase /tlb /nologo`
- To later on remove the registration use
  - `regasm.exe ComComponent.dll /u`

VBScript jobs calling COM components

VBScript job for Master

- This example works for a JobScheduler Master 32bit and 64bit
- The above COM component `ComComponent.dll` has to be registered for the respective architecture

Master runs VBScript that calls a COM component

```
<job  title="Master runs VBScript that calls a COM component" order="no" stop_on_error="no" tasks="1">
    <params/ >
    <script language="vbscript">
        <![CDATA[
        Set fso = CreateObject ("Scripting.FileSystemObject")
        Set stdout = fso.GetStandardStream (1)
        Set stderr = fso.GetStandardStream (2)

        Dim objTest, intResult
        Set objTest = CreateObject ("ComComponent.ComClassExample")
        intResult = objTest.AddTheseUp (100, 200)

        stdout.WriteLine "running VBScript job: " & intResult
        ]]>
    </script>
    <run_time />
</job>
```

VBScript Job for Agent

- This example works for a 32bit Agent
- The above COM component `ComComponent.dll` has to be registered for the 32bit architecture
- Basically the job script is the same as for the Master, consider use of the language `scriptcontrol:vbscript` and the assignment of a process class that points to an Agent.

Agent runs VBScript that calls a COM component

```
<job  title="Agent runs VBScript that calls a COM component" order="no" stop_on_error="no" tasks="1"
process_class="Agent">
    <params />
    <script  language="scriptcontrol:vbscript">
        <![CDATA[
        Set fso = CreateObject ("Scripting.FileSystemObject")
        Set stdout = fso.GetStandardStream (1)
        Set stderr = fso.GetStandardStream (2)

        Dim objTest, intResult
        Set objTest = CreateObject ("ComComponent.ComClassExample")
        intResult = objTest.AddTheseUp (100, 200)

        stdout.WriteLine "running VBScript job: " & intResult
        ]]>
    </script>
    <run_time />
</job>
```

PowerShell jobs calling COM components

PowerShell Job for Agent

- This example works for Agents with a JVM 32bit and 64bit.
- The above COM component `ComComponent.dll` has to be registered for the respective architecture.
- For details on the use of the JobScheduler API see PowerShell Jobs.

Agent runs PowerShell that calls a COM component

```
<job  title="Agent runs PowerShell that calls a COM component" order="no" stop_on_error="no" tasks="1"
process_class="Agent">
    <params />
    <script  language="powershell">
        <![CDATA[
$objTest = New-Object -ComObject "ComComponent.ComClassExample"
$intResult = $objTest.AddTheseUp(100, 200)

echo "running PowerShell job: $intResult"
        ]]>
    </script>
    <run_time />
</job>
```