ActiveBatch® Integrated Jobs Library Job Steps for

Microsoft® Windows PowerShell®

ActiveBatch®

Automate and Integrate PowerShell Scripts with ActiveBatch®

Benefits

- Automatically Trigger PowerShell Scripts to ensure workflows are executed reliably, improving efficiency throughout the organization.
- Integrate PowerShell Scripts within jobs and processes that span across various databases, applications and technologies of your IT environment.
- Improve Control and Flexibility of Workflows with event triggers, monitoring, alerts, audits and more.

Microsoft Windows PowerShell: IT Boundaries Identified

Microsoft PowerShell is the de facto standard scripting language for the automation and administration of systems and applications on Microsoft Windows. For IT professionals and programmers who use PowerShell to execute various tasks throughout an organization several major issues remain:

- How are these scripts triggered for execution?
- Is there a way to determine whether they have executed properly?
- Is there a way to establish a relationship between different but related PowerShell scripts?

Automate PowerShell Scripts and Use Them Within Workflows

ActiveBatch® Workload Automation provides integration and automation capabilities for PowerShell scripts. With ActiveBatch you can trigger script execution across a wide variety of events (e.g. file, email, web services, etc.) including business date/time, create workflows and ensure that these workflows execute reliably across your entire enterprise. With ActiveBatch's Integrated Jobs Library you can use the templated, drag-and-drop Job Steps to integrate PowerShell scripts and cmdlets without the need for custom script creation. This creates the best of both worlds in that you can avoid scripting entirely when needed.

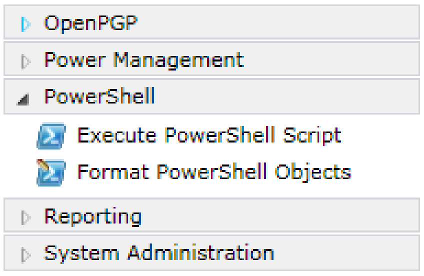
Object-Level Integrated Support

ActiveBatch provides object-level integrated support for PowerShell and offers object collection passing from one Job Step to another, providing users with improved integration and execution of PowerShell scripts. By leveraging PowerShell within the ActiveBatch environment, users can take advantage of more than 130 production-ready Job Steps to incorporate PowerShell scripts within workflows that contain other important business and/or administrative functionality. As PowerShell scripts are executed within the context of an ActiveBatch job, they can be enhanced by all the features of that object, including constraints, resource management, date/time or event-based triggers and more.

Use Case

Instead of manually triggering a PowerShell script to take a server offline, run an update and wait for it to complete, and then reboot the server for each server in an organization, ActiveBatch can automate this process by using the PowerShell script(s) in a workflow, ActiveBatch will automatically execute the PowerShell script for each server called, based on the parameters indicated.

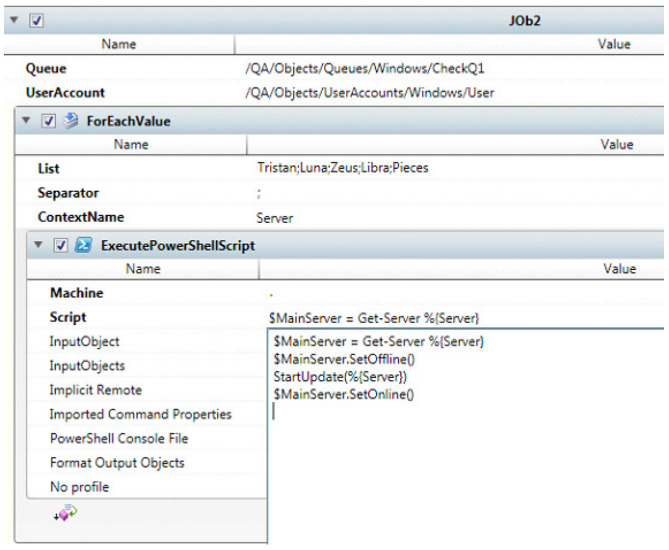
ActiveBatch: Version 8 SP3 and above



- OpenPGP
- Power Management
- PowerShell
  - Execute PowerShell Script
  - Format PowerShell Objects
- Reporting
- System Administration

ActiveBatch® Workflow for Microsoft Windows PowerShell

As displayed in the ActiveBatch Integrated Jobs Library



JOb2

Name	Value
Queue	/QA/Objects/Queues/Windows/CheckQ1
UserAccount	/QA/Objects/UserAccounts/Windows/User

ForEachValue

Name	Value
List	Tristan;Luna;Zeus;Libra;Pieces
Separator	;
ContextName	Server

ExecutePowerShellScript

Name	Value
Machine	.
Script	$MainServer = Get-Server %{Server}
InputObject	
InputObjects	
Implicit Remote	
Imported Command Properties	
PowerShell Console File	
Format Output Objects	
No profile	

```
$MainServer = Get-Server %{Server}
$MainServer.SetOffline()
StartUpdate(%{Server})
$MainServer.SetOnline()
```

Advanced Systems Concepts, Inc.

www.ActiveBatch.com