Recast

Retrieve package information

Last Modified on 04.16.26

Synopsis

This script will show the package information.
[Package Name> Snapshot Type > Actionset Type > Action Type] Key = Value

It will use the default PowerShell credential prompt if the username or passwords aren't provided within the script.

Sample script

```
[CmdletBinding()]
Param
(

 [Parameter(Mandatory = $true)]
 [URI]$uri = "",

 [Parameter()]
 [string]$packagename = "",

 [Parameter()]
 [string]$username = "",

 [Parameter()]
 [string]$password = "",

 [ValidateSet("Development", "Test", "Acceptance", "Production")]
 [string]$Type = "Production"

)

# Load the Powershell module
Import-Module "C:\Program Files (x86)\Liquit Workspace\PowerShell\3.0\Liquit.Server.PowerShell.dll" -Prefix "RecastSo
ftware"

# Set credentials
if ($username -eq "") {
 $credentials = Get-Credential
} else {
 $credentials = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $username, (Co
nvertTo-SecureString -String $password -AsPlainText -Force)
}

if ($packagename -eq "") {
 throw "No packagename set"
}

function Write-LiquitInfo {

 param (
  [array]$type,
  $object,
  $message
 )

 Write-Host "[" -NoNewline -ForegroundColor Blue
```

```
 $first = $false

 foreach ($item in $type) {
  if ($first) {
   Write-Host " > " -NoNewline -ForegroundColor Blue
  }
  Write-Host $item -NoNewline -ForegroundColor White
  $first = $true
 }

 Write-Host "]" -NoNewline -ForegroundColor Blue
 Write-Host " " -NoNewline
 Write-Host $object -NoNewline
 Write-Host " = " -NoNewline
 Write-Host $message

}

# Connect to the Application Workspace.
Connect-LiquitWorkspace -URI $uri -Credential $credentials | Out-Null

# Get all packages.
$package = Get-LiquitPackage | Where-Object { $_.name -eq $packagename }

## Show all Package Attributes
foreach ($item in $package.PSObject.Properties) {
 if ($item.Name -eq "ID") { continue }
 Write-LiquitInfo -type $package.Name -object $item.Name -message $item.Value
}

## Loop all Snapshots
foreach ($Snapshot in Get-LiquitPackageSnapshot -Package $package -Type $Type) {

 Write-LiquitInfo -type @($package.Name, $Snapshot.Type) -object "Type" -message $Snapshot.Type

 ## Loop All ActionSet
 foreach ($ActionSet in Get-LiquitActionSet -Snapshot $Snapshot ) {

  ## Show all ActionSet Attributes
  foreach ($item in $ActionSet.PSObject.Properties) {
   if ($item.Name -eq "ID") { continue }
   Write-LiquitInfo -type @($package.Name, $Snapshot.Type, $ActionSet.Type) -object $item.Name -message $item.V
alue
  }

  ## Loop All Action
  foreach ($Action in Get-LiquitAction -ActionSet $ActionSet | Sort-Object Position) {

   ## Show all Action Attributes
   foreach ($item in $Action.PSObject.Properties) {

    switch ($item.Name ) {

     "ID" { continue }

     "Settings" {

      ## change how Settings are displayed
      $settingsString = "@{"
      foreach ($setting in $Action.Settings.Keys) {
       $settingsString += $setting + ' = ''' + $($Action.Settings[$setting]) + '''; '
      }
```

```
    $settingsString += "}"

    Write-LiquitInfo -type @($package.Name, $Snapshot.Type, $ActionSet.Type, $Action.Type) -object $item.Name -m
essage $settingsString

   }

   Default {
    Write-LiquitInfo -type @($package.Name, $Snapshot.Type, $ActionSet.Type, $Action.Type) -object $item.Name -m
essage $item.Value
   }

  }

 }

}

}
```

Downloads

Script

> This sample is not supported by Recast. It was contributed by a community member and is published "as is." It seems to have worked for at least one person and might work for you. But please be sure to test, test, test before you do anything drastic with it.