Recast

Packages

Last Modified on 2026-05-25

Get-LiquitPackage

Synopsis

This command displays a list of all packages defined within the Application Workspace zone, or you can select a specific package.

Syntax

```
Get-LiquitPackage
  [-LiquitContext <LiquitContext>]
  [<CommonParameters>]

Get-LiquitPackage
  [-EntityRef] <EntityRef[]>
  [-LiquitContext <LiquitContext>]
  [<CommonParameters>]

Get-LiquitPackage
  [-Type] <string[]>
  [-LiquitContext <LiquitContext>]
  [<CommonParameters>]

Get-LiquitPackage
  [[-Group] <Group[]>]
  [-LiquitContext <LiquitContext>]
  [<CommonParameters>]

Get-LiquitPackage
  [[-User] <User[]>]
  [-LiquitContext <LiquitContext>]
  [<CommonParameters>]
```

Examples

This command displays a table with all packages, with their ID and name columns:

```
Get-LiquitPackage | Select id, name
```

This command selects all packages of the web type:

```
Get-LiquitPackage -type Web
```

This command selects a specific package:

```
Get-LiquitPackage -id 00000000-0000-0000-0000-000000000000
```

Often, you need a package ID, for example, when working with PowerShell or when calling a package via ShellAPI. You can use the following command:

```
Get-LiquitPackage -Name "RecastSoftware - 7-Zip (x64)".id
```

New-LiquitPackage

Synopsis

This command creates a new package.

Syntax

```
New-LiquitPackage
  [-Name] <string>
  [-Type] <string>
  [-DisplayName <string>]
  [-Excerpt <string>]
  [-Description <string>]
  [-Notes <string>]
  [-Website <string>]
  [-Priority <int>]
  [-Enabled <bool>]
  [-Offline <bool>]
  [-AutoLaunchable <bool>]
  [-Featured <bool>]
  [-Web <bool>]
  [-Icon <Content>]
  [-LiquitContext <LiquitContext>]
  [-WhatIf]
  [-Confirm]
  [<CommonParameters>]
```

Example

This command creates a new package with the name "PowerShellPackage", of the "Web" type. The package will be available without an Agent.

```
  New-LiquitPackage -name PowerShellPackage -type Web -Web $true -icon (New-LiquitContent -path C:\path\file.jpg
)
```

Parameters

Name	Value	Description	Required	Default value
Name	`<string>`	Provide a name for the package.	Yes	
Type	`<string>`	Provide a type for the package. Although this is a string value, the following types have custom images: • Install • Custom • Launch • Remote • Remote Control • Web	Yes	
DisplayName	`<string>`	The name displayed in the **Catalog** for the user.	No	

Name	Value	Description	Required	Default value
Excerpt	<string>	A shorter version of the application description, displayed on the application tiles.		
Description	<string>	Provide a description for the package.	No	
Notes	<string>	Fill in the internal notes as needed.		
Website	<string>	Provide users a website associated with the application.		
Priority	<int>	Determines the execution order when multiple packages are assigned to an event (on a user or device entitlement); the order of execution is from low to high.	No	1000
Enabled	<bool>	Determines whether or not the package is enabled.	No	True
Offline	<bool>	Determines whether or not the package is available in offline mode.	No	False
AutoLaunchable	<bool>	When enabled, users can choose to start this package on logon by enabling the Autolaunch option in the context menu of the app.	No	True
Featured	<bool>	When enabled, users can choose to start this package on logon by enabling the Autolaunch option in the context menu of the app.	No	False
Web	<bool>	Determines whether or not a package is available without an Agent.	No	False
Icon	<content>	Upload an icon to the package.	No	
LiquitContext	<LiquitContext>	Determines the selected zone.	No	Default

Set-LiquitPackage

Synopsis

This command lets you edit the properties of a package.

Syntax

```
Set-LiquitPackage
   [-Package] <Package[]>
   [-Name <string>]
   [-Type <string>]
   [-DisplayName <string>]
   [-Excerpt <string>]
   [-Description <string>]
   [-Notes <string>]
   [-Website <string>]
   [-Priority <int>]
   [-Enabled <bool>]
   [-Offline <bool>]
   [-AutoLaunchable <bool>]
   [-Featured <bool>]
   [-Web <bool>]
   [-Icon <Content>]
   [-LiquitContext <LiquitContext>]
   [-WhatIf]
   [-Confirm]
   [<CommonParameters>]
```

Example

The following script updates an existing package and renames it "PowerShellPackage"; users will see this package as "PowerShell Package" in the **Catalog** or **Workspace** tabs. The description of the package will be "Description":

```
$package = Get-LiquitPackage -id 00000000-0000-0000-0000-000000000000
   Set-LiquitPackage -Package $package -Type Launch -Name PowerShellPackage -DisplayName "PowerShell Package
" -Description "Description" -Enabled $true
```

Parameters

Name	Value	Description	Required	Default value
Name	`<string>`	Provide a name for the package.	Yes	
Type	`<string>`	Provide a type for the package. Although this is a string value, the following types have custom images: • Install • Custom • Launch • Remote • Remote Control • Web	Yes	
DisplayName	`<string>`	Provide a display name for the package.	No	
Excerpt	`<string>`	A shorter version of the application description, displayed on the application tiles.		
Description	`<string>`	Provide a description for the package.	No	
Notes	`<string>`			
Website	`<string>`	Provide users a website associated with the application.		
Priority	`<int>`	Determines the execution order when multiple packages are assigned to an event (on a user or device entitlement); the order of execution is from low to high.		

Recast

Name	Value	Description	Required	Default value
Enabled	<bool>	Determines whether or not the package is enabled.	No	True
Offline	<bool>	Determines whether or not the package is available in offline mode.	No	False
AutoLaunchable	<bool>	When enabled, users can choose to start this package on logon by enabling the Autolaunch option in the context menu of the app.		
Featured	<bool>	When enabled, users can choose to start this package on logon by enabling the Autolaunch option in the context menu of the app.		
Web	<bool>	Determines whether or not a package is available without an Agent.	No	False
Icon	<content>	Upload an icon to the package.	No	
LiquitContext	<LiquitContext>	Determines the selected zone.	No	Default

Copy-LiquitPackage

Synopsis

This command copies a package.

Syntax

```
Copy-LiquitPackage
  [-Snapshot] <PackageSnapshot>
  [-Name] <string>
  [-Publish {Development | Test | Acceptance | Production}]
  [-LiquitContext <LiquitContext>]
  [-WhatIf]
  [-Confirm]
  [<CommonParameters>]
```

Example

The following script creates a new package with a copy of the selected snapshot:

```
$package = Get-LiquitPackage -id 00000000-0000-0000-0000-000000000000
$snapshot = $package | Get-LiquitPackageSnapshot -id 00000000-0000-0000-0000-000000000000
  Copy-LiquitPackage -snapshot $snapshot -Name PowerShellPackage
```

Parameters

Name	Value	Description	Required	Default value
Snapshot	<PackageSnapshot>	The snapshot you wish to copy.	Yes	
Name	<string>	The name of the new package.	Yes	
Publish	{Development | Test | Acceptance | Production}	Choose the stage where you want to publish the snapshot.	No	Development

Recast

Name	Value	Description	Required	Default value
LiquitContext	<LiquitContext>	Determines the selected zone.	No	Default

Import-LiquitPackage

Synopsis

This command lets you import a resource and deliver it as a package.

Syntax

```
Import-LiquitPackage
   [-Resource] <Resource>
   [-Name] <string>
   [-Publish {Development | Test | Acceptance | Production}]
   [-Type {Default | Managed | Unmanaged}]
   [-LiquitContext <LiquitContext>]
   [-IgnoreMissingDependencies]
   [-Update]
   [-WhatIf]
   [-Confirm]
   [<CommonParameters>]
```

Example

The following script creates a new package with a copy of the selected snapshot:

```
$connector = Get-LiquitConnector -id 00000000-0000-0000-0000-000000000000
$resource = Get-LiquitResource -Connector $connector -id 00000000-0000-0000-0000-000000000000
Import-LiquitPackage -Resource $resource
```

Parameters

Name	Value	Description	Required	Default value
Resource	<Resource>	The resource to use as a basis for a new package or when updating one.	Yes	
Name	<string>	The alternative name for the package.	No	
Publish	{Development | Test | Acceptance | Production}	Choose the stage where you want to publish the snapshot.	No	Development
Type	{Default | Managed | Unmanaged}	The type of package to be created: • Default – it takes the type you set in **Connectors** > **Overview** > **Default package type**. • Managed – the package is managed by the connector; several options cannot be changed in order to allow easy updates. • Unmanaged – the package will be created based on the resource and you can make any changes to it.	No	Default

Name	Value	Description	Required	Default value
IgnoreMissingDependencies		Ignore missing dependencies on creation/update.	No	
Update		Update an existing package with the same name, otherwise an error is generated if a package with the same name exists.	No	
LiquitContext	<LiquitContext>	Determines the selected zone.	No	Default

Remove-LiquitPackage

Synopsis

This command removes a package from the Application Workspace.

Syntax

```
Remove-LiquitPackage
[-Package] <Package[]>
[-LiquitContext <LiquitContext>]
[-WhatIf]
 [-Confirm]
[<CommonParameters>]
```

Example

The following script removes the selected package from the Application Workspace:

```
$package = Get-LiquitPackage -id 00000000-0000-0000-0000-000000000000
Remove-LiquitPackage -package $package
```

Parameters

Name	Value	Description	Required	Default value
Package	<Package[]>	The package you wish to remove.	Yes	
LiquitContext	<LiquitContext>	Determines the selected zone.	No	Default