# Overview

This document contains reference material about the Azure Sphere command-line utilities:

* The **cutil** cloud utility provides commands that query or change the cloud service state.
* The **dutil** device utility provides commands to query or change device state for an attached Azure Sphere device.
* The **iutil** image utility packages an application image for sideloading or deployment to the Azure Sphere device.

The **cutil** and **dutil** command-line utilities support a common syntax:

***utility-name command operation [flags]***

In general, command is a noun and operation is a verb, so that the combination identifies both an action and the object of the action. Most commands and operations have both a full name and an abbreviation. For example, the **imageset** command is abbreviated **ims**.

The **iutil** utility uses only a command and omits the option. None of the syntax elements are case-sensitive.

Most flags have both a long name and an abbreviation. On the command line, introduce the long name with two hyphens and the abbreviation with a single hyphen. For example, the following two commands are equivalent:

***dutil wifi --ssid MyNetwork --key mynetworkkey***

***dutil wifi -s MyNetwork -k mynetworkkey***

Some flags accept more than one value, such as a list of filenames. You can either specify the flag once for each item in the list, or you specify the flag only once and separate the values in the list with commas and no intervening spaces. For example, these two commands have the same effect:

***iutil package -m image\_metadata.json -i bin -o myimage.imagepackage -x bin/app -x bin/custom***

***iutil package -m image\_metadata.json -i bin -o myimage.imagepackage -x bin/app,bin/custom***

# cutil cloud utility

The **cutil** cloud utility manages devices and deployments in the cloud. It provides the following commands:

|  |  |
| --- | --- |
| **COMMAND** | **DESCRIPTION** |
| **component, com** | Creates and manages components and images in the cloud. |
| **device, dev** | Manages devices in the cloud. |
| **devicegroup, dg** | Creates and manages device groups in the cloud. |
| **feed, f** | Creates and manages feeds in the cloud. |
| **ims, imageset** | Creates and manages image sets in the cloud. |
| **login** | Provides login to the Azure Sphere tenant. By default, all cutil commands apply to the current user’s AAD login identity and tenant. The login command lets you use a different identity. |
| **sku** | Creates and manages SKUs in the cloud. |

**Syntax**

cutil <command> <operation> <flags>

**Universal flags**

Universal flags can be used with any **cutil** command.

|  |  |
| --- | --- |
| **FLAG** | **DESCRIPTION** |
| -v, --verbose | Provides verbose output. |
| -?, --help | Displays help on the command. |

**Tip**: Many **cutil** operations require an ID or GUID that identifies the element (device, component, device group, feed, image set, or SKU) on which to operate. When only one such ID is required, operations typically use -i as the abbreviation, regardless of the long name of the associated flag.

**Component**

The component command creates and manages components and images in the cloud.

|  |  |
| --- | --- |
| **OPERATION** | **DESCRIPTION** |
| addimage, addimg | Uploads a new image and adds it to a component. |
| Create | Creates a new component. |
| getimage, getimg | Downloads an existing image from the cloud. |
| getimagemetadata, getimgmeta | Downloads the metadata for an existing image. |
| getimagestatus, getimgstatus | Gets the signing status of an existing image. |
| List | Lists all components in the current tenant. |
| Publish | Uploads a new image, adds it to a new image set, and adds the image set to an existing feed. |

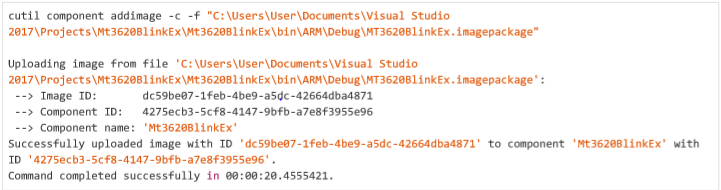
**addimage**

The **addimage** operation uploads a new image to the cloud and adds it to a component. The component ID is not required because it is part of the image metadata. Use the --autocreatecomponent (-c) flag to create the component if it does not already exist.

**Flags**

|  |  |
| --- | --- |
| -c, --autocreatecomponent | Creates a new component for the image if one does not already exist. |
| -f, --filepath path | Specifies the path to the image file to upload. Required. |

**Example**



**create**

The **create** operation creates a new component, given a component ID and a name.

**Flags**

|  |  |
| --- | --- |
| -i, --componentid GUID | Specifies the GUID of the component. This value appears in the ComponentUid field of the image\_metadata.json file for the application. Required. |
| -n, --name string | Specifies a name for the component. Required. |

**Example**



**getimage**

The **getimage** operation downloads a copy of an image that has already been added to a component.

**Flags**

|  |  |
| --- | --- |
| -f, --filepath path | Specifies the path and filename to which to save the image. The path can be relative to the current directory. Required. |
| -i, --imageid GUID | Specifies the image ID of the image to download. Required. |

**Example**



**getimagemetadata**

The **getimagemetadata** operation downloads the metadata for an image that has already been added to a component.

**Flags**

|  |  |
| --- | --- |
| -i, --imageid GUID | Specifies the GUID of the image for which to return the metadata. Required. |

**Example**



**getimagestatus**

The **getimagestatus** operation returns the signing status for an image that has already been added to a component.

**Flags**

|  |  |
| --- | --- |
| -i, --imageid GUID | Specifies the GUID of the image for which to return the status. Required. |

**Example**



**list**

The **list** operation lists all components in the current tenant.

**Example**



**publish**

The **publish** operation "publishes" a new image set for an existing component to an existing feed. It uploads a new image, adds it to a new image set, and adds the new image set to an existing feed. It combines the tasks of several other **cutil** commands:

* **cutil com addimg**
* **cutil imageset create**
* **cutil feed addimageset**

**Flags**

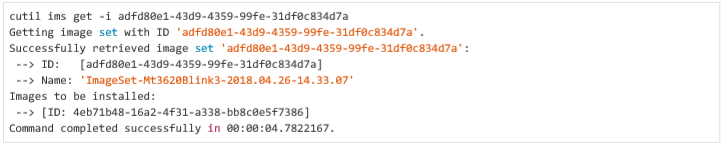
|  |  |
| --- | --- |
| -f, --feedid GUID | Specifies the GUID of the feed to which to add the image set. The feed must already exist and must deliver a component that has the same component ID as the specified image. Required. |
| -i, --imagepath filepath | Specifies the path and filename of the image to upload. The command auto-generates a component ID based on the information in the image package if a component with that ID does not already exist. Required. |
| -n, -newimagesetname string | Specifies a name for the new image set to be created. Optional. If omitted, a name is generated from the image metadata. |

**Example**

The following example uploads a new image for the existing Mt3620Blink3 component, creates an image set, and adds the image set to a feed that delivers the Mt3620Blink3 component.



The command did not include the optional --newimagesetname flag, so **cutil** creates a name that includes the filename of the image package and a timestamp. The **cutil imageset** get command shows the details for this image set:



**device**

The **device** command manages devices in the cloud.

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| **claim** | Claims a previously unclaimed device in the cloud. |
| **fieldprep, field** | Sets up a device for OTA updates by combining the tasks of several other cutil commands. |
| **get** | Gets details about a device from the cloud. |
| **getimageset, getims** | Returns information about the image sets to be deployed to the device. |
| **linktofeed,linkf** | Links a device to a feed |
| **setdevicegroup, setdg** | Moves the device into a device group. |
| **setsku** | Sets the product SKU for a device. |

**claim**

The **claim** operation associates a device with your Azure Sphere tenant. Before you claim a device, be sure that you are logged in to the tenant that you plan to use with Azure Sphere. See the login command for more information.

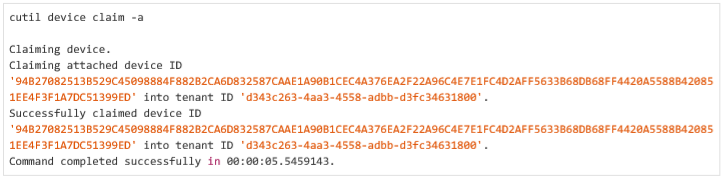
**Flags**

|  |  |
| --- | --- |
| -a, --attached | Claims the Azure Sphere device that is currently attached to the PC. |
| -i, --deviceid String | Claims the device that has the specified device ID. |

**Important**

*This operation requires either the --attached flag or the --deviceid flag.*

**Example**



**fieldprep**

The **fieldprep** operation sets up a device to receive OTA updates. It combines the work of several other cutil commands:

* cutil sku create
* cutil device setsku
* cutil devicegroup create
* cutil device setdevicegroup

The specific tasks that **fieldprep** performs depend on the whether a product SKU and device group have already been assigned. If the product SKU or the device group does not already exist, the command creates it and assigns it to the device, provided that a SKU name or a device group name is supplied with the appropriate flag. See Examples for details.

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| -a, --attached | Sets up the Azure Sphere device that is currently attached to the PC. Either --attached or --deviceid is required. |
| -r, --devicegroupid Guid | Sets the ID of the device group to which to add the device. If this flag is omitted, the command creates a new device group and enables application updates for the new group. Either --devicegroupid or -newdevicegroupname is required. |
| -i, --deviceid String | Specifies the Azure Sphere device to which the command applies. Either --attached or --deviceid is required. |
| -g, -newdevicegroupname String | Specifies a name for the new device group to create. Use this flag only if you have not already set up a device group for OTA updates. Whenever possible, use an existing device group to avoid cluttering the tenant with unneeded groups. Either --devicegroupid or --newdevicegroupname is required |
| -n, -newskunameString | Specifies a name for the new product SKU to create for the device. Use this flag only if you do not use the --skuid flag. If you already have a product SKU that is appropriate for this device, supply it with the --skuid flag to avoid cluttering your tenant with numerous SKUs that are only used once. If neither the --newskuname nor the --skuid flag is present, the command checks whether the device already has a product SKU and if so, uses it; otherwise, the command fails. |
| -d, --skudescription String | Specifies a string that describes the new product SKU to create. Ignored if the device already has a product SKU. |
| -s, --skuidGUID | Assigns an existing product SKU to the device. |

**Examples**

Example 1. Create a product SKU and device group for device

This example creates a new product SKU and a new device group, and assigns both to the attached device.

***cutil dev fieldprep -a --newdevicegroupname AppUpdateGroupPO --newskuname UtopiaPO***

As the output shows, the command creates a device group named AppUpdateGroupPO and a product SKU named UtopiaPO and assigns both to the attached device. The new device group is enabled for OTA loading of application updates.



Example 2. Assign existing product SKU and device group to device

This example assigns an existing product SKU and device group to the attached device.



Example 3. Assign device to different device group

In the next example, the device already has a product SKU and is part of a device group. Here the **fieldprep** operation changes the device group to which the device belongs. This command is useful for testing device groups that contain many devices, or for moving a device from a development environment that does not enable OTA application updates to a production environment that does.



**get**

The **get** operation returns information about a device.

**Flags**

|  |  |
| --- | --- |
| -a, --attached | Returns information about the Azure Sphere device that is currently attached to the PC. |
| -i, --deviceid String | Returns information about the device that has the specified device ID. |

**Important**

*This operation requires either the --attached flag or the --deviceid flag*.

**Example**



**getimageset**

The **getimageset** operation returns information about the current image set for a device. The current image set is the image set that will be deployed to the device the next time an OTA update occurs. It is not necessarily the image set that is currently running on the device.

**Flags**

|  |  |
| --- | --- |
| -a, --attached | Returns information about the current image set for the Azure Sphere device that is currently attached to the PC. |
| -f, --full | Lists all image sets that will be installed on the device. By default, only application image sets are listed. |
| -i, --deviceid String | Returns information about the current image set for the Azure Sphere device that has the specified device ID. |

**Important**

*This operation requires either the --attached flag or the --deviceid flag.*

**Example**



**linktofeed**

The **linktofeed** operation provides an abbreviated way to deploy an application. It creates a feed and assigns that feed to the devices in the same device group as a specified device. It combines the tasks of the following **cutil** commands:

* cutil component addimage
* cutil feed create
* cutil devicegroup addfeed
* cutil imageset create
* cutil feed addimageset

**Flags**

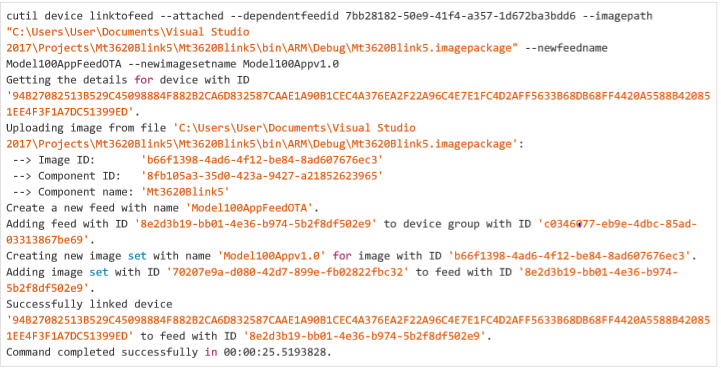
|  |  |
| --- | --- |
| -a, --attached | Selects the device group to which the attached the Azure Sphere device belongs. Either the --attached flag or the -deviceid flag is required. |
| -c, --componentid GUID | Specifies the component ID of the application. If you use the --newfeedname flag to create a new feed, you must also supply either the --componentid or the --imagepath flag. |
| -d, -dependentfeedid GUID | Specifies the ID of the dependent feed for the application. Application feeds depend on the latest Azure Sphere OS feed, which is currently named Preview MT3620 release Feed. Required when creating a new feed. |
| -i, --deviceid String | Selects the device group for the Azure Sphere device with the specified device ID. Either the --attached flag or the -deviceid flag is required. |
| -f, --feedid GUID | Specifies the ID of the feed to link to the device group for the specified device. If you omit this flag, cutil creates a new feed and assigns it the friendly name that you specify with the optional --newfeedname flag. If you already have a feed for this component, use it to avoid cluttering your tenant with redundant feeds. Either the --feedid flag or the -newfeedname flag is required. |
| -p, --imagepath String | Specifies the path to the image package to upload. If you omit this flag, the feed is created but is not assigned an initial image set. Optional. |
| -n, -newfeedname GUID | Specifies a friendly name for the new feed. Use only if you do not specify a feed ID with the --feedid flag. If you use the --newfeedname flag to create a new feed, you must also supply either the --componentid or the --imagepath flag. Either the --feedid flag or the --newfeedname flag is required. |
| -s, -newimagesetname String | Specifies a friendly name for the new image set that the command creates when you use the --imagepath flag. If you specify an image path but not an image set name, cutil automatically generates a name for the image set. |

**Examples**

***Important***

***Do not copy the dependent feed ids from the examples.*** *Always use the* ***cutil feed list*** *command to list the feeds in your tenant, and then copy the feed ID from the latest Preview MT3620 Feed. This ensures that the device continues to operate properly with the correct version of the Azure Sphere OS and that the software is not inadvertently overwritten by a version that is not compatible with the latest tools and Azure Sphere Security Service.*

The first example creates a new feed, a new component, and a new image set. The new feed depends on the Preview MT3620 feed and is named BlinkLink.



The next example creates a new feed for an existing component. The feed is dependent on an Preview MT3620 feed and is named BlinkLink. The feed services the devices in the same device group as the attached device.

***cutil dev linktofeed -a -c 16995a70-377f-4bd2-b29d-1b0fffcbe287 -n BlinkLink --dependentfeedid 7bb28182-50e9-41f4-a357-1d672ba3bdd6***



The **cutil feed get** command shows the resulting feed. No image sets are associated with the feed because the preceding **cutil dev linktofeed** command did not specify an image path. You can later add an image set to the feed by using the **cutil feed addims** command.



**setdevicegroup**

The **setdevicegroup** operation moves a device to a device group.

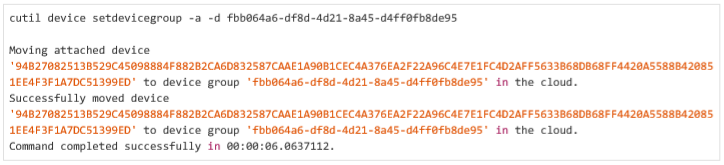
**Flags**

|  |  |
| --- | --- |
| -a, --attached | Sets the device group for the Azure Sphere device that is currently attached to the PC. |
| -d, --devicegroup GUID | Specifies the device group to which to move the device. Required. |
| -i, --deviceid String | Sets the device group for the Azure Sphere device with the specified device ID. |

**Important**

*This operation requires either the --attached flag or the --deviceid flag.*

**Example**



**setsku**

The **setsku** operation sets the product SKU for a connected device that contains a Azure Sphere device.

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| -a, --attached | Sets the product SKU for the Azure Sphere device that is currently attached to the PC. |
| -i, --deviceid String | Sets the product SKU for the Azure Sphere device with the specified device ID. |
| -s, -SkuIdGUID | Specifies the product SKU to set for the device. Use cutil sku create to create a product SKU and cutil sku list to list the SKUs in the current tenant. |

**Important**

*This operation requires either the --attached flag or the --deviceid flag.*

**Example**



**devicegroup**

The **devicegroup** command manages device groups in the cloud.

|  |  |
| --- | --- |
| **OPERATION** | **DESCRIPTION** |
| **addfeed, addf** | Adds a feed to a device group. |
| **create** | Creates a device group. |
| **get** | Returns information about a device group. |
| **list** | Lists all device groups. |
| **listdevices, listdev** | Lists all devices in a device group. |
| **listfeeds, listf** | Lists all feeds that are assigned to a device group. |

**addfeed**

The **addfeed** operation assigns a feed to a device group.

**Flags**

|  |  |
| --- | --- |
| -f, --feedid GUID | Specifies the feed to assign to the device group. |
| -i, --devicegroupid GUID | Specifies the device group to which to add the feed. |

**Example**



**create**

The create operation creates a new device group and assigns it a friendly name.

Device group names are only for human use and need not be unique. Codename 4x4 services identify device groups by their IDs.

By default, application software updates are enabled for all device groups, so that devices receive OTA deployments of application software automatically. You can change this default by specifying the --noapplicationupdates (-a) flag when you create a group. Disabling updates means that the devices in the group will not receive OTA updates and must instead be updated by sideloading, either through Visual Studio or by using the dutil sideload command.

Flags

|  |  |
| --- | --- |
| -a, -noapplicationupdates | Disables application updates for this device group. |
| -n, --name String | Specifies an alphanumeric name for the device group. If the name includes embedded spaces, enclose it in quotation marks. Required. |

**Examples**



get

The **get** operation returns information about a device group.

**Flags**

|  |  |
| --- | --- |
| -i, --devicegroupid GUID | Specifies the GUID that identifies the device group. Required. |

**Example**



**list**

The **list** operation lists all device groups in the current tenant.

**Example**



**listdevices**

The **listdevices** operation lists each device in a device group, along with any SKUs that are assigned to the device.

**Flags**

|  |  |
| --- | --- |
| -i, --devicegroupid GUID | Specifies the GUID that identifies the device group. Required. |

**Example**



**listfeeds**

The **listfeeds** operation lists all feeds that are assigned to a device group.

**Flags**

|  |  |
| --- | --- |
| -i, --devicegroupid GUID | Specifies the GUID that identifies the device group. Required. |

**Example**



**Feed**

The **feed** command creates and manages feeds, which deliver software updates.

|  |  |
| --- | --- |
| **OPERATION** | **DESCRIPTION** |
| **addimageset, addims** | Adds an image set to a feed. |
| **create** | Creates a feed. |
| **get** | Returns information about a feed. |
| **list** | Lists all feeds. |
| **listdevicegroups, listdg** | Lists all device groups that a feed targets. |
| **listimagesets, listims** | Lists all image sets that are assigned to a feed. |

**addimageset**

The **addimageset** operation adds an image set to a feed. The image set must represent the component ID that was specified with the cutil feed create command.

**Flags**

|  |  |
| --- | --- |
| -i, --feedid GUID | Specifies the ID of the feed to which to add the image set. Required. |
| -s, --imagesetid GUID | Specifies the ID of the image set to add to the feed. Required. |

**Example**



**create**

The **create** operation creates a new feed.

**Flags**

|  |  |
| --- | --- |
| -s, --chipskuid GUID | Specifies one or more chip SKU IDs that this feed targets. You can either use this flag multiple times to specify multiple SKUs or use the flag once and separate multiple SKU IDs with commas and no intervening spaces. Required. |
| -c, --componentid GUID | Specifies the ID of the component that this feed delivers. The component must already have been created by either the **cutil com create** or **cutil com addimage --autocreatecomponent** command. Required. |
| -f, -dependentfeedid GUID | Specifies the ID of the Azure Sphere OS feed on which this feed depends. Currently, applications must specify the Preview MT3620 release Feed for the most recent Azure Sphere OS release. To get a list of system software feeds and IDs, use the **cutil feed list** command. Required. |
| -n, --name String | Specifies an alphanumeric name for the feed. Required |
| -p, --productskuid GUID | Specifies one or more product SKU IDs that this feed targets. You can either use this flag multiple times to specify multiple SKUs or use the flag once and separate multiple SKU IDs with commas and no intervening spaces. Required. |

**Examples**

**Important**

*Do not copy the dependent feed ids from the examples. Always use the cutil feed list command to list the feeds in your tenant, and then copy the feed ID from the latest Preview MT3620 Feed. This ensures that the device continues to operate properly with the correct version of the Azure Sphere OS and that the software is not inadvertently overwritten by a version that is not compatible with the latest tools and Azure Sphere Security Service.*



**get**

The **get** operation returns information about a feed.

**Flags**

|  |  |
| --- | --- |
| -i, --feedid GUID | Specifies the GUID that identifies the feed. Required. |

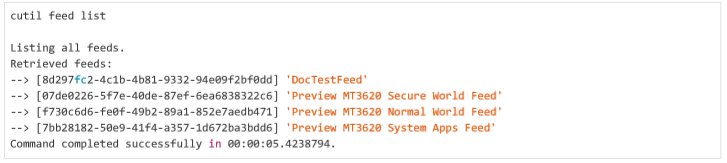
**Example**



**list**

The **list** operation lists all feeds in the current tenant.

**Example**



**listdevicegroups**

The **listdevicegroups** operation lists all device groups to which a feed is assigned.

**Flags**

|  |  |
| --- | --- |
| -i, --feedid GUID | Specifies the GUID that identifies the feed. Required. |

**Example**



**listimagesets**

The **listimagesets** operation lists all image sets that are assigned to a particular feed.

**Flags**

|  |  |
| --- | --- |
| -i, --feedid GUID | Specifies the GUID that identifies the feed. Required. |

**Example**



**imageset**

The imageset command creates and manages image sets.

|  |  |
| --- | --- |
| **OPERATION** | **DESCRIPTION** |
| **create** | Creates an image set. |
| **get** | Returns information about an image set. |
| **list** | Lists all image sets in the current tenant. |

**create**

The **create** operation creates a new image set.

**Flags**

|  |  |
| --- | --- |
| -m, -imageid GUID | Specifies one or more image IDs that identify the images that the image set contains. You can either use this flag multiple times to specify multiple images or use the flag once and separate multiple image IDs with commas and no intervening spaces. Currently, image sets for applications can include only one image. Required. |
| -n, --name String | Supplies an alphanumeric name for the image set. Required. |

**Example**



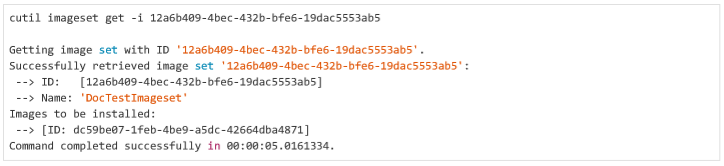
**get**

The **get** operation returns information about an image set.

**Flags**

|  |  |
| --- | --- |
| -i, --imagesetid GUID | Specifies the GUID that identifies the image set. Required. |

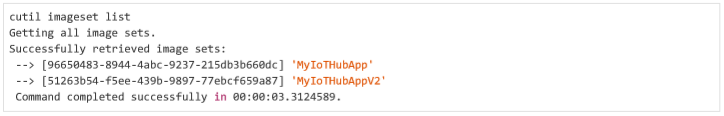
**Example**



**list**

The **list** operation lists all image sets in the current tenant.

**Example**



**login**

The **login** option provides login to Codename 4x4 services.

When you use **cutil**, the Codename 4x4 services verify your identity by using Microsoft Azure Active Directory (AAD). AAD uses Single Sign-On (SSO), which typically defaults to an existing identity on your PC. If this identity is not valid for use with Codename 4x4 services, **cutil** commands may fail.

The **cutil login** command enables you to sign in explicitly to Codename 4x4 services. Upon success, Codename 4x4 services use this identity for subsequent **cutil** commands. In most cases, you should only have to sign in once. Example

**Example**

*cutil login*

In response, you should see a dialog box that lists your credentials or prompts you to log in. If the list includes the identity that you use for Codename 4x4 services, choose that identity. If not, enter the appropriate credentials.

**sku**

The **sku** command creates and manages SKUs.

|  |  |
| --- | --- |
| **OPERATION** | **DESCRIPTION** |
| **create** | Creates a SKU. |
| **get** | Returns information about a SKU. |
| **list** | Lists all SKUs. |

**create**

The **create** operation creates a new product SKU and associates a friendly name with it.

**Flags**

|  |  |
| --- | --- |
| -d, --description String | Provides a description of the SKU. |
| -n, --name SkuName | Supplies an alphanumeric name for the SKU. SKU names are case sensitive. Required. |

**Examples**



**get**

The **get** operation displays information about a SKU.

**Flags**

|  |  |
| --- | --- |
| -i, --skuid GUID | Specifies the GUID that identifies the SKU. Required. |

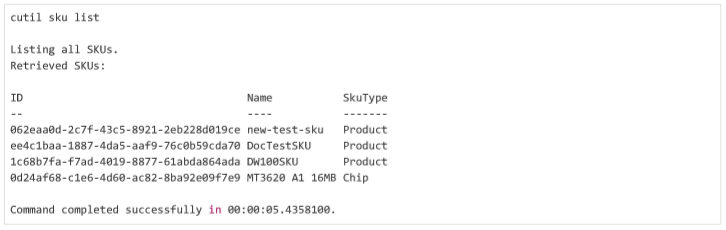
**Example**



**list**

The **list** operation displays all the SKUs and names that are associated with the current tenant.

**Example**



# dutil device utility

The **dutil** device utility provides commands to manage the configuration and operation of your device.

|  |  |
| --- | --- |
| **COMMAND** | **DESCRIPTION** |
| **device** | Displays the status or properties of the device. |
| **recover** | Replaces the system software onto the device. |
| **sideload** | Loads an application image on the device, or changes the status of the current application. |
| **wifi** | Adds, removes, modifies, or displays the Wi-Fi configuration for the device. |

**Syntax**

*dutil <command> <option> <flags>*

**Universal flags**

|  |  |
| --- | --- |
| -?, --help | Displays help on the command. |
| -v, --verbose | Provides verbose output. |

All **dutil** commands return a zero status to indicate success and a non-zero status to indicate failure.

**device**

The **device** command returns information about the device.

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| id | Displays the Azure Sphere device ID. |
| sdkversion | Displays the version of the Azure Sphere SDK on the device. |

**id**

The **id** option returns the Azure Sphere device ID.

**Example**

*dutil device id*

*Device ID: 50D90F3CD12007688D04346AB753B958C337E2E13BC5BC05BF5026B7DEE5C4013957A86F6976FED2A90DEAC7287780B919180DDDC44C79 19932DB615924C2BBC*

**sdkversion**

The **sdkversion** option returns the version of the Azure Sphere SDK that is currently on the device.

**Example**

*dutil device sdkversion*

*SDK version: 4.0.0+93882*

**recover**

The **recover** command loads new system software onto the device.

**Flags**

|  |  |
| --- | --- |
| -b, --bootloader filename | Specifies the filename of the recovery bootloader to use. By default, the command uses recoveryloader.bin from the folder named with the --images flag. |
| -c, --capability filename | Specifies the filename of the device capability image to apply to the device. For Microsoft use only. |
| -i, --images folder | Specifies the path to a folder that contains the Microsoft-supplied image packages to write to the device. By default, recovery uses the recovery-loader.bin and recover.imagemanifest files from this folder. Required. |
| -m, --manifest filename | Specifies the path to the recovery image manifest file. By default, the command uses the recovery.imagemanifest file from the folder specified with the --images flag. |

**Example**

*dutil recover -i .\recovery*

**sideload**

The **sideload** command provides options to manage the application on the device. It has the following options:

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| delete | Deletes the current application from the device. |
| deploy | Loads an application onto the device. |
| start | Starts the application that is loaded on the device. |
| status | Returns the status of the current application on the device. |
| stop | Stops the application that is running on the device. |

**delete**

The **delete** option deletes the current application from the device. After you delete the application, application status is notPresent. If you have an OTA deployment, cloud services will then load the deployed image onto your device.

**Example**

***dutil sideload delete***

*dutil sideload status*

*No app present.*

**deploy**

The **deploy** option sideloads an application image onto the device, but does not start it. To start the application, use the start option.

**Flags**

|  |  |
| --- | --- |
| -p, --imagepackage filename | Specifies the filepath to the application to deploy. Required. |

**Example**

*dutil sideload deploy -p BlinkEx.imagepackage*

**start**

The **start** option starts the application that is currently on the device.

**Flags**

|  |  |
| --- | --- |
| -d, --debug | Starts the application in debug mode. |

**Example**

*dutil sideload start*

**status**

The **status** option shows the status of the application.

**Example**

*dutil sideload status*

*App state: running*

**stop**

The **stop** option stops the application that is currently on the device. It does not unload or delete the application.

**Example**

***dutil sideload stop***

*dutil sideload status*

*App state: stopped*

**wifi**

The **wifi** command changes the wireless configuration for the device. To use the device on a wireless network, you must add information about the network and enable the network on the device. If your application uses the WifiConfig API, you must also include the WifiConfig capability in the application's app\_manifest.json file.

|  |  |
| --- | --- |
| **OPTION** | **DESCRIPTION** |
| add | Adds the details of a wireless network to the device. |
| disable | Disables a wireless network on the device. |
| enable | Enables a wireless network on the device. |
| list | Lists the current Wi-Fi configuration for the device. |
| remove | Removes the details of a wireless network from the device. |
| scan | Scans for available networks. |
| status | Displays the status of the wireless interface. |

**add**

The **add** option adds information about a Wi-Fi connection to the device. A device can have multiple Wi-Fi connections.

**Flags**

|  |  |
| --- | --- |
| -k, --key key | Specifies the WPA/WPA2 key for the new network. Omit to add this SSID as an open network. |
| -s, --ssid ssid | Specifies the SSID of the network. Network SSIDs are case-sensitive. Required. |

**Example**

***dutil wifi add -s MyNetwork -k myKey123***

*Add network succeeded:*

*ID : 2*

*SSID : MyNetwork*

*Configuration state : enabled*

*Connection state : unknown*

*Security state : psk*

**disable**

The **disable** option disables Wi-Fi on a specified network.

**Flags**

|  |  |
| --- | --- |
| -i, --id id | Specifies the ID of the network. Required. |

**Example**

***dutil wifi disable -i 2***

*Successfully disabled network:*

*ID : 2*

*SSID : MyNetwork*

*Configuration state : disabled*

*Connection state : disconnected*

*Security state : psk*

**enable**

The **enable** option enables Wi-Fi on a specified network.

**Flags**

|  |  |
| --- | --- |
| -i, --id id | Specifies the ID of the network. Required. |

**Example**

***dutil wifi enable -i 2***

*Successfully enabled network:*

*ID : 2*

*SSID : MyNetwork*

*Configuration state : enabled*

*Connection state : connected*

*Security state : psk*

**list**

The **list** option displays information about all the Wi-Fi connections on the device.

**Example**

***dutil wifi list***

*Network list:*

*ID : 0*

*SSID : NETGEAR21-5G*

*Configuration state : enabled*

*Connection state : connected*

*Security state : psk*

*ID : 1*

*SSID : Netgear21*

*Configuration state : disabled*

*Connection state : disconnected*

*Security state : psk*

*ID : 2*

*SSID : MyNetwork*

*Configuration state : enabled*

*Connection state : disconnected*

*Security state : psk*

**remove**

The **remove** option removes information about a wireless network from the device.

**Flags**

|  |  |
| --- | --- |
| -i, --id id | Specifies the ID of the network. Required. |

**Example**

***dutil wifi remove -i 0***

*Successfully deleted network:*

*Success.*

**scan**

The **scan** option scans for wireless networks within range of the device.

**Example**

***dutil wifi scan***

*Scan results:*

*SSID : NETGEAR21*

*Security state : psk*

*BSSID : 44:94:fc:36:c8:65*

*Signal level : -66*

*Frequency : 2442*

*SSID : CenturyLink9303*

*Security state : psk*

*BSSID : 58:8b:f3:09:ae:d2*

*Signal level : -75*

*Frequency : 2412*

*SSID : NETGEAR21-5G*

*Security state : psk*

*BSSID : 44:94:fc:36:c8:64*

*Signal level : -86*

*Frequency : 5765*

*SSID : belkin.c32*

*Security state : psk*

*BSSID : 08:86:3b:0b:cc:32*

*Signal level : -86*

*Frequency : 2462*

**status**

The **status** option displays information about the current Wi-Fi connection on the device. **Example**

***dutil wifi status***

*SSID : NETGEAR21*

*Configuration state : enabled*

*Connection state : connected*

*Security state : psk*

*Frequency : 2442*

*Mode : station*

*Key management : WPA2-PSK*

*WPA State : COMPLETED*

*IP Address : 192.168.1.15*

*MAC Address : 52:cf:ff:3a:76:1b*

# iutil image utility

The **iutil** image utility packages and manages images. This utility is useful for developers who build applications without using Visual Studio. If you use the Visual Studio Tools Preview for Azure Sphere to build applications, the tools create image packages for you.

|  |  |
| --- | --- |
| **COMMAND** | **DESCRIPTION** |
| package | Packages a compiled and linked image with a metadata file to create an executable Azure Sphere application. |

**Syntax**

*iutil <command> <flags>*

**Universal flags**

|  |  |
| --- | --- |
| -?, --help | Displays help on the command. |
| -v, --verbose | Provides verbose output. |

**package**

The **package** command creates an executable application from a compiled and linked image and a metadata file.

**Flags**

|  |  |
| --- | --- |
| -i, --input path | Identifies the input directory, which **iutil** uses as the system root for the Azure Sphere image file. Required. |
| -m, --metadata file | Identifies the metadata file (in JSON format) to include in the package. Required. |
| -o, --output file | Specifies a filename for the output image package. Required. |
| -x, --executables executable1, executable2… | Subpaths to one or more files to mark as executable in the image package. By default, Windows files are not executable when packaged into an image. The subpaths are relative to the path that was specified with the -i flag. The paths can use either Windows filename syntax (backslashes) or Linux filename syntax (forward slashes); spaces, commas, and semicolons are not allowed. You can either supply the -x flag for each executable file or use the flag only once and supply multiple paths, separated by commas without intervening spaces. Note that Visual Studio automatically renames the binary file for the application to /bin/app, so the Visual Studio project name can include punctuation without causing any errors. Required. |

**Example**

*iutil package -m image\_metadata.json -i bin -o myimage.imagepackage -x bin/app*