Tools

Last updated by | Paul Kelleher | 17 Jun 2020 at 07:49 GMT

Azure Command Line Tools (AZ CLI)

Contents

Preparation

List Available Extensions

az extension list-available -o table

Name	Version	Summary	Preview	Installed
aks-preview	0.3.2	Provides a preview for upcoming AKS features	True	False
alias	0.5.2	Support for command aliases	True	False
azure-devops	0.5.0	Tools for managing Azure DevOps.	True	False
etc	etc			

Update Extensions

command	features
az extension add --name azure-devops	
az extension add --name interactive	
az extension add --name storage-preview	

az feature list

az feature list --query [?state=="NotRegistered"].name | grep Storage

```
"Microsoft.DBforMariaDB/newStorageLimit",
  "Microsoft.DBforMySQL/newStorageLimit",
  "Microsoft.DBforPostgreSQL/newStorageLimit",
  "Microsoft.RecoveryServices/PremiumStorageBackup",
  "Microsoft.Storage/AllowADFS",
  "Microsoft.Storage/AllowArchive",
  "Microsoft.Storage/AllowHNS",
  "Microsoft.Storage/AllowPreReleaseRegions",
  "Microsoft.Storage/AllowStorageV1Accounts",
  "Microsoft.Storage/AllowValidationRegions",
  "Microsoft.Storage/armApiPreviewAccess",
  "Microsoft.Storage/BlobIndex",
  "Microsoft.Storage/BlobQuery",
  "Microsoft.Storage/CustomerControlledFailover",
  "Microsoft.Storage/EncryptionAtRest",
  "Microsoft.Storage/LivesiteThrottling",
  "Microsoft.Storage/premiumblob",
  "Microsoft.Storage/version",
  "Microsoft.Storage/Versioning",
  "Microsoft.Storage/XArchive",
  "Microsoft.Storage/AllowPremiumFiles",
  "Microsoft.Storage/Tags"
```

az feature register

az feature register --namespace Microsoft.Storage --name Tags

Standard Commands

az account show

```
{
  "environmentName": "AzureCloud",
  "id": "09f4d59a-6e2c-4368-ab93-4ad53bf96fa4",
  "isDefault": true,
  "name": "VBL Core AAD Dev & Test",
  "state": "Enabled",
  "tenantId": "1f8a9b14-7258-40a6-8643-f8d4f174c58e",
  "user": {
    "cloudShellID": true,
    "name": "vdi@vcapv3.onmicrosoft.com",
    "type": "user"
  }
}
```

az account list

```
[
  {
    "cloudName": "AzureCloud",
    "id": "09f4d59a-6e2c-4368-ab93-4ad53bf96fa4",
    "isDefault": true,
    "name": "VBL Core AAD Dev & Test",
    "state": "Enabled",
    "tenantId": "1f8a9b14-7258-40a6-8643-f8d4f174c58e",
    "user": {
      "cloudShellID": true,
      "name": "vdi@vcapv3.onmicrosoft.com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "id": "09f4d59a-6e2c-4368-ab93-4ad53bf96fa4",
    "isDefault": true,
    "name": "VBL Core AAD Dev & Test",
    "state": "Enabled",
    "tenantId": "1f8a9b14-7258-40a6-8643-f8d4f174c58e",
    "user": {
      "cloudShellID": true,
      "name": "vdi@vcapv3.onmicrosoft.com",
      "type": "user"
    }
  }
]
```


Custom Scripts

az-sub

```
#!/bin/bash
export list=$(az account list --query '[].{Id:id,Name:name}' -o table | grep VBL| tr ' ' '_' | tr '\t' ':'
if [[ $1 = --help ]]
        then
        printf '\n
select azure subscription
                \n'
mapfile -d £ -t azsubs < <(printf $list)


printf '\n
VBL Azure : Select Subscription
------------------------------\n\n'
az account show -o table

printf '\nswitch subscription? Y/N : ' ; read -r change

if [ $change = n ] || [ $change = N ] ; then printf '\n subscription unchanged\n\n'; exit 0; fi

for i in "${!azsubs[@]}"; do
            printf "%s) %s\n" "$i" "${azsubs[$i]}"
    done
    printf ' \n'
    printf 'Select a subscription from the above list: '
    IFS= read -r opt
    if [[ $opt =~ ^[0-9]+$ ]] && (( (opt >= 0) && (opt <= "${#azsubs[@]}") )); then
            selection=$(printf ${azsubs[$opt]} | sed 's/_/ /g' | awk '{print $1}')
            az account set --subscription $selection
            printf '\n'
            az account show -o table
            printf '\n'
        else
                    printf 'bad\n'
            fi
```

set-storage-acct

```
#!/bin/bash
export FILE_SHARE="networx"
export ACCOUNT_NAME="vblnetworxdata"
```

az-dir

```
#!/bin/bash

source ./set-storage-account

RESOURCE_NAME=$ACCOUNT_NAME
ID=$(az resource list --query "[?contains(name, '${RESOURCE_NAME}')].[id]" -o tsv)
export CONNECTION_STRING=$(az storage account show-connection-string  --id $(az resource list --query "[?cc


function help(){
        printf '

        pk-custom azcopy (az storage) helper)
        ------------------------------------

        1. this script expects one argument, which should be a filename including its path (eg ./foldername
        2. this script will not execute if the file does not exist
        3. this script will check the fileshare in the storage account named and will check if the full fil
        4. this script will upload the file exactly replicating the folder structure supplied. it is recomm
        5. this script will log all runs

        az-ul ./foldername/filename.ext

        '

        exit 0
}


if ! [[ $# == 1 ]];
then
                help
fi

PATHNAME=$1

## Strip start and ending slashes
[[ $PATHNAME == ?/* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}
[[ $PATHNAME == /*/ ]] && PATHNAME=${PATHNAME:1:(-1)}
[[ $PATHNAME == */ ]] && PATHNAME=${PATHNAME:0:(-1)}
[[ $PATHNAME == /* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}

function check_storage_dir_exists(){

        DIRCOUNT=$(echo $PATHNAME | awk -F/ '{print NF}')

        for i in $(seq 1 $DIRCOUNT)
        do
                export CHECKPATH=$(echo ${PATHNAME} | cut -d/ -f 1-$i)
                echo $i $CHECKPATH

                if ! [[ $(az storage directory exists  --account-name $ACCOUNT_NAME --share-name $FILE_SHAR
                then
                az storage directory create --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connect
                fi
        done
}

check_storage_dir_exists

if [[ $1 == "/" ]]
then
        az storage file list --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $CON
else
        az storage file list --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $CON
fi
```

az-ls

```
#!/bin/bash

source ./set-storage-account

RESOURCE_NAME=$ACCOUNT_NAME
ID=$(az resource list --query "[?contains(name, '${RESOURCE_NAME}')].[id]" -o tsv)
export CONNECTION_STRING=$(az storage account show-connection-string  --id $(az resource list --query "[?co

PATHNAME=$1
## Strip start and ending slashes
[[ $PATHNAME == ?/* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}
[[ $PATHNAME == /*/ ]] && PATHNAME=${PATHNAME:1:(-1)}
[[ $PATHNAME == */ ]] && PATHNAME=${PATHNAME:0:(-1)}
[[ $PATHNAME == /* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}

if [[ $# == 0 ]]
then
        az storage file list --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $ACC
else
        az storage file list --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $ACC
fi
```

upload-file

```
#!/bin/bash

source ./set-storage-account

RESOURCE_NAME=$ACCOUNT_NAME
ID=$(az resource list --query "[?contains(name, '${RESOURCE_NAME}')].[id]" -o tsv)
export CONNECTION_STRING=$(az storage account show-connection-string  --id $(az resource list --query "[?co



function help(){
        printf '

        pk-custom azcopy (az storage) helper)
        ------------------------------------

        1. this script expects one argument, which should be a filename including its path (eg ./foldername
        2. this script will not execute if the file does not exist
        3. this script will check the fileshare in the storage account named and will check if the full fil
        4. this script will upload the file exactly replicating the folder structure supplied. it is recomm
        5. this script will log all runs

        az-ul ./foldername/filename.ext

        '

        exit 0
}


if ! [[ $# == 1 ]];
then
        if [[ $2 == '--replace' ]]
        then
                REPLACE="True"
        else
                help
        fi
fi


FILENAME=$(basename ${1})
PATHNAME=$(dirname ${1})

## Strip start and ending slashes
[[ $PATHNAME == ?/* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}
[[ $PATHNAME == /*/ ]] && PATHNAME=${PATHNAME:1:(-1)}
[[ $PATHNAME == */ ]] && PATHNAME=${PATHNAME:0:(-1)}
[[ $PATHNAME == /* ]] && PATHNAME=${PATHNAME:1:$(echo ${PATHNAME} | wc -c)}

function check_storage_dir_exists(){

        DIRCOUNT=$(echo $PATHNAME | awk -F/ '{print NF}')
        echo "Checking directories"
        for i in $(seq 1 $DIRCOUNT)
        do
                export CHECKPATH=$(echo ${PATHNAME} | cut -d/ -f 1-$i)

                if ! [[ $(az storage directory exists  --account-name $ACCOUNT_NAME --share-name $FILE_SHAR
                then
                az storage directory create --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connect
                fi
        done
}

function check_file_exists() {
        if [[ $(az storage directory exists --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connect
        then
                echo " "
                echo " a directory with the same name as the file exists "
                echo " please choose another filename or rename the directory"
                echo " "
                exit 0
```

```
        fi

        if [[ $(az storage file exists  --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-
        then
                FILEDETAILS=$(az storage file show  --account-name $ACCOUNT_NAME --share-name $FILE_SHARE -
                echo " "
                echo " File Exist   size  / last modified"
                echo "Azure File : "$FILEDETAILS
                echo "Local File : " $(ls -lrth 2019/jan/pkdate  | awk '{print $5" "$6" "$7" "$8}')
                echo " "
                echo " if you wish to overwrite, please state --override after the filename"
                echo " "
        else
                check_storage_dir_exists
                upload_file
        fi
}

function upload_file(){
        az storage file upload --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $C
        az storage file url --account-name $ACCOUNT_NAME --share-name $FILE_SHARE --connection-string $CONN
}

check_file_exists
```

Azure CloudShell (Info) (Open) | Storage Explorer | Azure Data Studio | Azure Resources Explorer | ARM Visualizer | Azure Calculator | Office 365 |Azure Portal