az-pyproxy

This Python package provides a proxy class for calling az commands directly from Python.

Requirements

- azure cli
- Python 3.x

Installation

Via PIP

```
pip3 install azpyproxy
```

or

```
python3 -m pip install azpyproxy
```

Building from source

```
wget https://github.com/tomasvotava/az-pyproxy/archive/master.zip
unzip master.zip
cd az-pyproxy-master
python setup.py build
python setup.py install
```

Usage

```python
from azure_pyproxy import Azure
az = Azure()

# command: az vm start virtual
az.vm_start("virtual")

# additional flags and parameters
# next method call expands like this:
# az resource list --resource-group="YOUR_RESOURCE_GROUP"
az.resource_list(resource_group="YOUR_RESOURCE_GROUP")

# positional arguments are simply concatenated after the command call, so commands can be also passed as arguments
```

```
az.vm("deallocate", name="virtual", resource_group="YOUR_RESOURCE_GROUP")
# expands as:
# az vm deallocate --name="virtual" --resource_group="YOUR_RESOURCE_GROUP"

# by default, underscores in flag names are replaced with dashes
az.vm_deallocate(name="virtual", resource_group="YOUR_RESOURCE_GROUP")
# az vm deallocate --name="virtual" --resource_group="YOUR_RESOURCE_GROUP"

# however, this can be overriden by specifying replace_underscore=False
az.vm_deallocate(name="virtual", resource_group="YOUR_RESOURCE_GROUP",
replace_underscore=False)
# az vm deallocate --name="virtual" --resource_group="YOUR_RESOURCE_GROUP"

# the command returns json output from the commands, if you would like to redirect
stdout/stderr/stdin instead, just pass keep_output=True
az.vm_deallocate(name="virtual", resource_group="YOUR_RESOURCE_GROUP",
keep_stdout=True)
# Returns True/False based on errorcode of the command, prints all output
```

Default subscription and resource group

If you do not want to specify subscription/resource_group in each call and you did not set your defaults using az command or Azure web UI, you can use classmethods designed to deal with it for you.

```
from azure_pyproxy import Azure

az = Azure()

# Shows your default subscription
az.account_show()

az.account_show(subscription="another_subscription")
# Shows your "another_subscription"

Azure.set_subscription("another_subscription")
az.account_show()
# Shows your "another_subscription"

# If you have set your subscription using Azure.set_subscription
# and you need to use few calls using another one,
# you can always specify your subscription as a parameter (this overrides default
setting)
Azure.set_subscription("another_subscription")
az.account_show(subscription="yet_another_one")
# Shows info for "yet_another_one"

# In case you want to switch back to your az default, you can reset your
subscription:
Azure.reset_subscription()
```

All of the above works with resource groups as well:

```python
from azure_pyproxy import Azure
az = Azure()

Azure.set_resource_group("resource-group")

az.vm_start(name="virtual")
```

Code above starts VM named `virtual` in resource group `resource-group`. Following code resets the setting for you:

```python
Azure.reset_resource_group()
```