Prepare your environment for the Azure CLI

Article • 05/10/2025

In this tutorial step, you execute your first Azure CLI commands! This includes finding and setting your active subscription and setting default values. You also learn how to create resource groups containing a random ID to always guarantee a unique resource name.

If you don't have an Azure subscription, create an Azure free account before you begin.

To complete this tutorial step, you need `contributor` or higher permissions on your subscription.

Install the Azure CLI

Get started by first selecting your environment.

- Use the Bash environment in Azure Cloud Shell by selecting the **Open Cloud Shell** button in the top right corner of each Azure CLI code block.
- If you prefer to run the Azure CLI reference commands locally, install the Azure CLI.

The current version of the Azure CLI is **2.73.0**. For information about the latest release, see the release notes. To find your installed version and see if you need to update, run az version.

Sign in to Azure using the Azure CLI

There are several authentication options when working with the Azure CLI. The Azure CLI's default authentication method for logins uses a web browser and access token to sign in.

1. Run the `az login` command.

   Azure CLI

   ```
   az login
   ```

   If Azure CLI can open your default browser, it initiates authorization code flow and opens the default browser to load an Azure sign-in page.

   Otherwise, it initiates the device code flow and instructs you to open a browser page at https://aka.ms/devicelogin . Then, enter the code displayed in your terminal.

If no web browser is available or the web browser fails to open, you can force device code flow with `az login --use-device-code`.

2. Sign in with your account credentials in the browser.

Find and change your active subscription

After logging into the Azure CLI, always check your current subscription. If you aren't working under the subscription you prefer, change your subscription using az account set. Here's a code example to use:

Azure CLI

```
# see your current/default subscription
az account show

# find the list of subscriptions available to you
az account list --output table

# change your current/default subscription
az account set --subscription <mySubscriptionName>

# you can also set your subscription using a subscription ID
az account set --subscription <00000000-0000-0000-0000-000000000000>
```

Console output for `az account show` command:

Output

```
{
  "environmentName": "AzureCloud",
  "homeTenantId": "00000000-0000-0000-0000-000000000000",
  "id": "00000000-0000-0000-0000-000000000000",
  "isDefault": true,
  "managedByTenants": [],
  "name": "Your storage account name",
  "state": "Enabled",
  "tenantId": "00000000-0000-0000-0000-000000000000",
  "user": {
    "name": "yourName@contoso.com",
    "type": "user"
  }
}
```

Learn about resource groups

A resource group is a container for Azure resources. To create a resource group, you need `contributor` permissions or higher in your subscription.

Create a resource group

1. Within a subscription, a resource group name must be unique. First check if the name you want is available using the az group exists command. An output value of `false` means that the name is available.

   Azure CLI

   ```
   az group exists --name <myUniqueRGname>
   ```

2. Retrieve a list of supported regions for your subscription with the az account list-locations command. For a list of available Azure locations, see Choose the right Azure region for you .

   Azure CLI

   ```
   az account list-locations --query "[].{Region:name}" --output table
   ```

3. It's time to create your resource group! Use the az group create command.

   Azure CLI

   ```
   az group create --location <myLocation> --name <myUniqueRGname>
   ```

Create a resource group containing a random ID

When testing, it's best to create a resource group that contains a random ID in the name. Adding a random ID to your resource group name allows you to retest your code without having to wait for a prior resource group of the same name to be removed from Azure.

Bash and PowerShell variable syntax is different. Copy the correct script for your environment.

Bash

Azure CLI

```
let "randomIdentifier=$RANDOM*$RANDOM"
location="eastus"
resourceGroup="msdocs-tutorial-rg-$randomIdentifier"
az group create --name $resourceGroup --location $location --output json
```

Bash and PowerShell console output:

Output

```
{
  "id": "/subscriptions/00000000-0000-0000-0000-
000000000000/resourceGroups/msdocs-tutorial-rg-000000000",
  "location": "eastus",
  "managedBy": null,
  "name": "msdocs-tutorial-rg-000000000",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```

Set environment variables

The Azure CLI offers several options to allow you to reuse common parameter values. These default values are stored in environment variables and are unique for each sign in.

1. Set your default resource group.

   Azure CLI

   ```
   az config set defaults.group=<msdocs-tutorial-rg-0000000>
   ```

2. Set multiple environment variables at once.

   Many Azure CLI parameters take multiple values separated by a space. Configuration values are one such instance. The next example sets both the `.location` and `.group` defaults that are used by the `--location` and `--resource-group` parameters of every Azure CLI command.

Azure CLI

```
az config set defaults.location=westus2 defaults.group=<msdocs-tutorial-
rg-0000000>
```

3. Set your default output.

   When you chose to work in Azure Cloud Shell, or install the Azure CLI locally, the default output is automatically set to `json`. However, this is one of the most important defaults to understand and set. **Output determines what appears on your console and what is written to your log file.** Always use an output of `none` when you're creating resources that return keys, passwords and secrets.

   Azure CLI

   ```
   az config set core.output=none
   ```

   In this tutorial, we aren't working with secrets. Set the default back to `json` so you can see the returned output of each reference command in this tutorial.

   Azure CLI

   ```
   az config set core.output=json
   ```

4. Learn to use `az init`.

   The Azure CLI has a reference command that walks you through configuring your environment. Type `az init` in your console and press **Enter**. Follow the prompts provided.

   Azure CLI

   ```
   az init
   ```

   The first nice thing about az init is that it gives you all of your current settings! Here's example output:

   Output

   ```
   Your current config settings:

     Output format: JSON
   ```

```
  [core.output = json]

  Standard error stream (stderr): All events
  [core.only_show_errors = false]

  Error output: Show recommendations
  [core.error_recommendation = on]

  Syntax highlighting: On
  [core.no_color = false]

  Progress Bar: On
  [core.disable_progress_bar = false]


Select an option by typing it's number

  [1] Optimize for interaction
      These settings improve the output legibility and optimize for human
interaction

  [2] Optimize for automation
      These settings optimize for machine efficiency

  [3] Customize settings
      A walk through to customize common configurations

  [4] Exit (default)
      Return to the command prompt

? Your selection:
```

5. Find and read your configuration file.

   If you work under a "trust but verify" mindset, you want to know where your configuration files are stored and what they contain. The configuration file itself is located at `$AZURE_CONFIG_DIR/config`. The default value of `AZURE_CONFIG_DIR` is `$HOME/.azure` on Linux and macOS, and `%USERPROFILE%\.azure` on Windows. Find your config file now and see what it contains.

Get more details

Do you want more detail on one of the subjects covered in this tutorial step? Use the links in this table to learn more.

Expand table

Subject	Learn more
Environments	Choose the right Azure command-line tool
Sign in options	Sign in with Azure CLI
Terms	Azure CLI terminology and support levels
Subscriptions	Manage subscriptions using the Azure CLI
Resource groups	Manage resource groups using the Azure CLI
Configurations	Configure the Azure CLI
Azure roles	Azure roles, Microsoft Entra roles, and classic subscription administrator roles

Next Step

Now that you've learned how to configure your environment, proceed to the next step to learn the scripting differences between Bash, PowerShell and Cmd.

Learn Azure CLI syntax differences in Bash, PowerShell and Cmd