Lab Guide: Variables

There are four ways to get, set, or use variables in Nerdio scripted actions:

1. Hard-coded variables
2. Built-in Nerdio environment variables
3. Secure variables, Inherited variables, and Environment variables
4. Runtime variables, AKA parameters

Hard-Coded Variables

Hard-coded variables are defined in your script. For example:

$MyVariable = ‘MyValue’

They should generally be defined near the top of your script, so that it is easy to edit them as needed. Every time you run the script, the same value will be used. To provide new values, you’ll need to edit the variable in the script, or clone the script and edit the copy.

Hard-coded variables are best for things like a download URL, which should rarely need to be changed. Consider using inherited variables instead (discussed below).

Nerdio Variables

Nerdio pre-defines some variables for you. These variables will be available to your script when it is run; you do not need to define them ahead of time. The variables will be defined based on the specific context in which the script is run; for example, when running a script on a VM, the value of the $AzureVMName variable will be the name of the VM running the script.

The Nerdio-defined variables are:

$HostPoolId
$HostPoolName
$AzureSubscriptionId
$AzureSubscriptionName
$AzureResourceGroupName

$AzureRegionName
$AzureVMName
$ADUsername (if passing AD credentials)
$ADPassword (if passing AD credentials)
$DesktopUser (if using with personal host pool)

Secure, Inherited, and Environment Variables

Secure variables are a way of passing sensitive information, such as API keys, passwords, or other secrets, to your scripts securely. To define the value of a secure variable, navigate in Nerdio Manager to Settings->Integrations.

Secure variables are provided in a hashtable available to your script. To retrieve the value of a secure variable from within a script, use the format $SecureVars.VariableName.

Inherited variables are defined at the MSP level and inherited at the account level. They can optionally be overridden at the account level. To define or override the value of an inherited variable, navigate in Nerdio Manager to Settings->Integrations at the MSP or account level.

Inherited variables are also defined in a hashtable available to your script. To use the value of an inherited variable inside a script, use the format $InheritedVars.VariableName

Environment Variables provide information specific to the environment in which the script is running, whether that is at the MSP level or an account. The environment variables available are:

CustomerName
DefaultDomain
SubscriptionId
TenantDomain
TenantId

To use an environment variable, use the hashtable notation $EnvironmentalVars.VariableName

Runtime variables (AKA Parameters)

Runtime variables are only available in Azure runbooks (for now) and allow you to create a script that expects certain parameters to be passed when the script is run. You define these variables in json notation within a comment block called Variables.

The variables definition block looks like this:

```
<# Variables:
{
  "Param1": {
    "Description": "The first parameter",
    "IsRequired": true
  },
  "Param2": {
    "Description": "An optional parameter",
    "IsRequired": false
  }
}
#>
```