Accelerate

“What’s new in Azure Infrastructure: Using Open Source and Other Technologies”

## Table of Contents

[Table of Contents 1](#_Toc452310075)

[Installing and using GIT as a repository 2](#_Toc452310076)

[Using ARM templates to further automate VM Deployments 6](#_Toc452310077)

[Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager 6](#_Toc452310078)

[Authentication 7](#_Toc452310079)

[Set the Resource Manager mode 7](#_Toc452310080)

[Create a resource group 7](#_Toc452310081)

[Use resource group templates 8](#_Toc452310082)

[Export a resource template 10](#_Toc452310083)

Installing and using GIT as a repository

In this next Module, we are going to install and configure GIT as a repository for ARM templates.

### **Install NODE.JS**

Download install the node.JS package from [https://nodejs.org](https://nodejs.org/).

When you install Node.js, you’ll want to ensure your PATH variable includes your install path so you can call Node from anywhere. Node comes with npm installed so you should have a version of npm.

### **Testing Your Install**

Create a new directory named hello-world, add a new app.js file with the following content:

/\* app.js \*/

console.log('Hello World!');

In the command prompt (or terminal on Mac) , run

$ node app.js

If you get any error regarding Node is not found, open a new command prompt (or terminal) to reflect the new environment variable for Node.

### **Installing and Setting up Git**

Chances are, you already have git running on your computer or you at least have heard of Git before. For this module, you will need to setup git locally and know few basics about Git.

**Getting Git for Mac**

There are several ways to install Git on a Mac. You can choose from one of the following ways:

1) If you want a more up to date version, you can also install it via a binary installer. An OSX Git installer is maintained and available for download at the Git website, at <http://git-scm.com/download/mac>.

2) If you already have Homebrew, you can install Git by executing:

$ brew install git

Make sure you update your $PATH environment variable to include the latest install path of Git. For example:

$ echo 'export PATH="/usr/local/bin:/usr/local/sbin:~/bin:$PATH"' >> ~/.bash\_profile

**Getting Git for Windows**

If you’re already using Chocolatey or Windows 10’s package manager to install software, you can simply run the following command from an elevated Powershell or Bash console (right click, select ‘Run as Administrator’):

$ cinst git.install

$ cinst poshgit

# Restart PowerShell / CMDer before moving on - or run

$ env:Path = [System.Environment]::GetEnvironmentVariable("Path","Machine") + ";" + [System.Environment]::GetEnvironmentVariable("Path","User")

$ cinst git-credential-winstore

$ cinst github

Verifying Git Installation

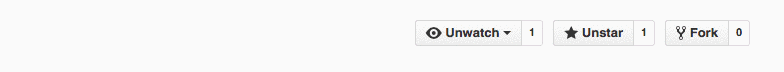
Now that you have Git installed, open up PowerShell on Windows or terminal on Mac. If everything worked correctly, you should be able to rungit --version.

Signing up for a Free GitHub account

Before we can get started, you need to register with GitHub for a free account. Either create or login into your account on [www.github.com](http://www.github.com).

Forking a Repository to your Account

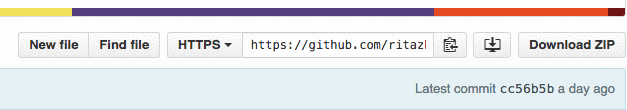
Now you are ready to do more with Git. Let’s start with a sample project. Head over to the Accelerate Lab repository on <https://github.com/robert-bakker/AccelerateLab> and click the little ‘fork’ button in the upper right.



This will create a copy of the repository as it exists in the original account into your own account.

Cloning the Repository to Your Machine

Visit your fork (**which should be at github.com/{your\_github\_username}/acceleratelab)** and copy the “HTTPS Clone URL”. Using this URL, you’re able to clone the repository, which downloads the whole repository, including its history and information about its origin locally. From PowerShell on Windows or terminal on Mac, change into the directory where you would like to clone your repo.



* Copy your repo URL

Clone the code to your local machine.

$ git clone https://github.com/{your\_github\_username}/acceleratelab

This should generate output that looks roughly like this:

$ git clone https://github.com/{your\_github\_username}/acceleratelab

Cloning into 'acceleratelab'...

...

...

Checking connectivity... done.

You can run explorer . from PowerShell on Windows or open . from Terminal on Mac to open up the folder in Explorer or Finder respectively. All the files are there - including the history of the whole repository. The connection to your fork ({your\_github\_username}/acceleratelab) is still there.

Creating a new Branch for your repo

In modern Git development, every single change that you want to make to the code base will be made in a “branch”. Like a tree branch, the branch is “based” on a different branch, and unlike other SCM systems, Git branches are very lightweight. The default branch name for GitHub repositories is master. In order to create a new branch, you can always run:

# This makes sure that your new branch is based on master

# When the default branch of a repo is "master", you should "git checkout master"

# The below command creates a new branch

$ git checkout -b my-new-branch

You can now go ahead and make your changes - adding files, writing code, fixing bugs. Keep in mind that a branch should host isolated changes. For example, you should create one branch that fixes a bug, then another branch for to develop a new feature you want to implement.

Staging your Changes for a Commit

Now that you made your changes, you can “stage” them for a commit. Whenever you stage a file for a commit, you make a snapshot of the file at the time you’re staging it for a commit. If you change a file after you staged it, you will have to stage it again. To stage a file, simply run:

$ git add ./path-to/my-file.md

If you just want to stage all files in your current repository (including deletions), run:

$ git add --all

Committing your Changes

Now that your changes have been staged, we’re ready to commit them. You can either pass the commit command a title for your commit - or omit the parameter, in which case Git will open up the default text editor for you to create a commit message.

To commit the quick way:

$ git commit -m "Add new feature: Git is Awesome"

To commit the long way, allowing you to define both title and message of your commit:

$ git commit

Pushing your new Branch to Your Fork on GitHub

Let’s say you have implemented a new feature, made some changes, committed the changes - now we have to make sure that your changes also end up on GitHub. To do so, we have to push your local branch to your fork on GitHub. Run the command below, using the name of the branch you want to push to

$ git checkout NAME\_OF\_YOUR\_NEW\_BRANCH

$ git push -u origin NAME\_OF\_YOUR\_NEW\_BRANCH

# Using ARM templates to further automate VM Deployments

## Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager

### Azure resources

Use the Azure Resource Manager to create and manage a group of resources (user-managed entities such as a virtual machine, database server, database, or website) as a single logical unit, or resource group.

One advantage of the Azure Resource Manager is that you can create your Azure resources in a declarative way: you describe the structure and relationships of a deployable group of resources in JSON templates. The template identifies parameters that can be filled in either inline when running a command or stored in a separate JavaScript Object Notation (JSON) parameters file.

This allows you to easily create new resources using the same template by simply providing different parameters. For example, a template that creates a website will have parameters for the site name, the region the website will be located in, and other common settings.

When a template is used to modify or create a group, a deployment is created, which is then applied to the group.

After you create a deployment, you can manage the individual resources imperatively on the command line, just like you do in the classic deployment model.

For example, use CLI commands in Resource Manager mode to start, stop, or delete resources such as [Azure Resource Manager virtual machines](https://github.com/Azure/azure-content/blob/master/articles/virtual-machines/virtual-machines-linux-cli-deploy-templates.md). https://github.com/Azure/azure-content/blob/master/articles/virtual-machines/virtual-machines-linux-cli-deploy-templates.md

## Authentication

Working with the Azure Resource Manager through the Azure CLI currently requires you to authenticate to Microsoft Azure by using the azure login command and then specifying an account managed by Azure Active Directory - either a work or school account (an organizational account) or a Microsoft account.

## Set the Resource Manager mode

Because the CLI is not in Resource Manager mode by default, use the following command to enable Azure CLI Resource Manager commands.

azure config mode arm

## Create a resource group

A resource group is a logical grouping of resources such as network, storage, and compute resources. Almost all commands in the Resource Manager mode need a resource group.

First delete the resource group created in the previous exercises.

* Important: all resources you previously created will be lost.

View all resource groups in your subscription

azure group list

Then delete the specific resource group:

azure group delete [resource group name]

Create a new resource group using the last 4 digits of your mobile phone number as a unique identifier

azure group create -n accelerate[uniqueid] -l westeurope

* You will deploy to this accelerate[ID] resource group later when you use a template to launch a Linux VM. Once you have created a resource group, you can add resources like virtual machines and networks or storage.

## Use resource group templates

When working with templates, you can either [create your own](https://github.com/Azure/azure-content/blob/master/articles/resource-group-authoring-templates.md) (https://github.com/Azure/azure-content/blob/master/articles/resource-group-authoring-templates.md), or use one of the community-contributed [Quick Start templates](https://azure.microsoft.com/documentation/templates/),( https://azure.microsoft.com/documentation/templates/) which are also available on [GitHub](https://github.com/Azure/azure-quickstart-templates) (https://github.com/Azure/azure-quickstart-templates).

### Creating a multi node VMM scale set running an Apache/PHP website.

To create this more advanced scenario (VM Scale Set, loadbalancer, Apache & PHP), using an azure resource template from GitHub we will use the [201-vmss-lapstack-autoscale](https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-lapstack-autoscale) template created by Madhan Arumugam Ramakrishnan, available on the Azure QuickStart templates repository on GitHub.

<https://github.com/Azure/azure-quickstart-templates/blob/master/201-vmss-lapstack-autoscale/>

This template creates a Simple self-contained Ubuntu/Apache/PHP (LAP stack) with auto scaling & load balancing. The VM Scale Set scales up when avg CPU across all VMs is larger than 60%, and it automatically scales down again when the average CPU is less than 50%. This template will deploy the following:

* Deploy the VM Scale Set with an instance count of 1
* After it is deployed look at the resource group public IP address resource (in portal or resources explorer). Get the IP or domain name.
* Browse to the website (port 80), which shows the current backend VM name.
* Hit the "Do work" button with an iteration count of say 300 (represents seconds of max CPU).
* After a few minutes the VM Scale Set capacity will increase, and refreshing the browser and going to the home page a few times will show additional backend VM name(s).

You can increase the work by connecting to more backend websites, or decrease by letting the iterations time-out, in which case the VM Scale Set will scale down - hence after about 10 minutes the capacity should be back down to 1.

### Deploy this template to your Azure resource group

You can use this template directly from [GitHub](https://github.com/Azure/azure-quickstart-templates), instead of downloading one to your computer. To do this, pass the URL to the azuredeploy.json file for the template in your command by using the **--**template**-uri** option. To get the URL, open azuredeploy.json on GitHub in raw mode, and copy the URL that appears in the browser's address bar. You can then use this URL directly to create a deployment by using a command similar to the following.

* (the new resource group needs to be created first using: azure group create -n accelerate[uniqueid] -l westeurope)

azure group deployment create [Resource Group Name] [Resource Group Deployment Name] --template-uri [RAW Github link to azuredeploy.json]

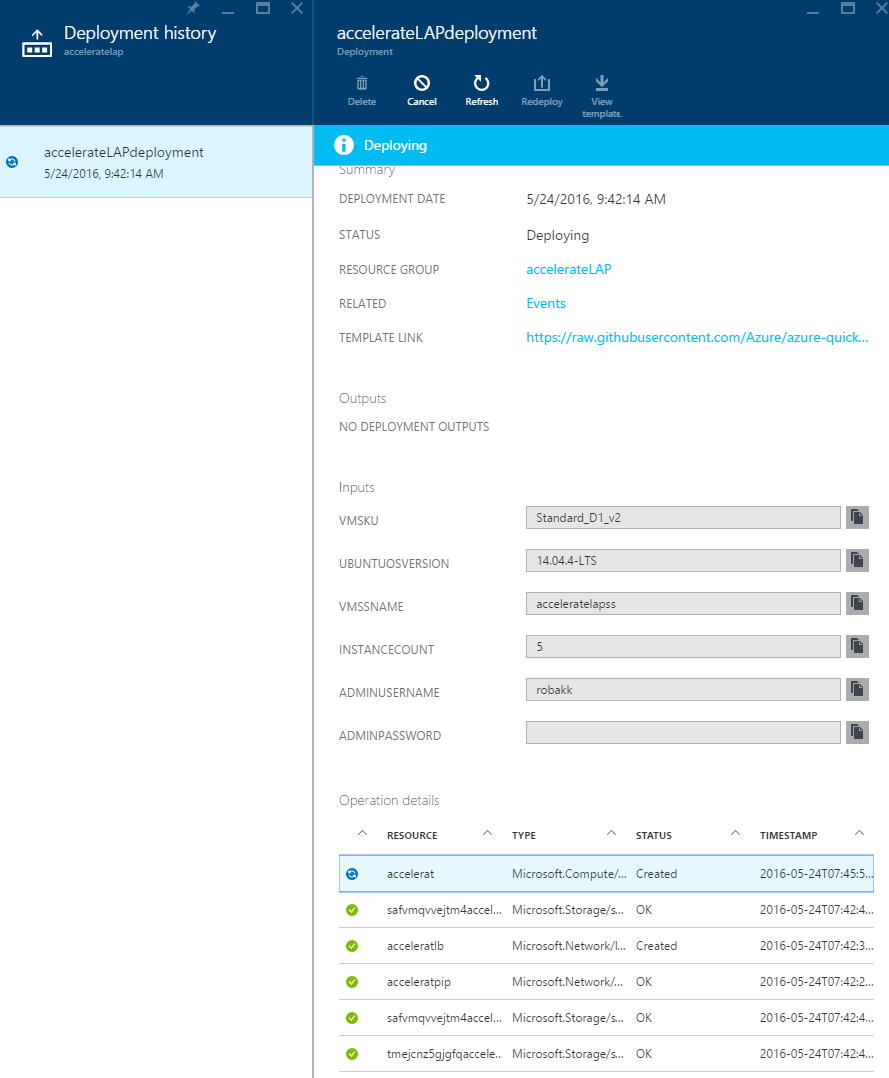
azure group deployment create accelerateLAP accelerateLAPdeployment https://raw.githubusercontent.com/robert-bakker/AccelerateLab/master/LabFiles/201-vmss-lapstack-autoscale/azuredeploy.json

You are then prompted to enter the necessary template parameters.

* vmssName : the name of the VM Scale Set
* instancecount : how many VMs need to be in the scale set
* adminusername : user name for the administrator
* Password: the admin password

This deployment will take approximately 5 minutes. You can watch the progress of the deployment on portal.azure.com in the resource group specified:

Now you should be able to use a web browser to connect to [scalesetname].westeurope.cloudapp.azure.com and enter a number of seconds to



## Export a resource template

For an existing resource group, you can view the Resource Manager template for the resource group. Exporting the template offers two benefits:

1. You can easily automate future deployments of the solution because all of the infrastructure is defined in the template.
2. You can become familiar with template syntax by looking at the JSON that represents your solution.

Using the Azure CLI, you can either export a template that represents the current state of your resource group, or download the template that was used for a particular deployment.

* **Export the template for a resource group** - This is helpful when you have made changes to a resource group, and need to retrieve the JSON representation of its current state. However, the generated template contains only a minimal number of parameters and no variables. Most of the values in the template are hard-coded. Before deploying the generated template, you may wish to convert more of the values into parameters so you can customize the deployment for different environments.

To export the template for a resource group to a local directory, run the azure group export command as shown in the following example. (Substitute a local directory appropriate for your operating system environment.)

azure group export accelerate[uniqueid] ~/azure/templates/

* **Download the template for a particular deployment** -- This is helpful when you need to view the actual template that was used to deploy resources. The template will include all of the parameters and variables defined for the original deployment. However, if someone in your organization has made changes to the resource group outside of what is defined in the template, this template will not represent the current state of the resource group.

To download the template used for a particular deployment to a local directory, run the azure group deployment template download command.

azure group deployment template download TestRG testRGDeploy ~/azure/templates/downloads/

Template export is in preview, and not all resource types currently support exporting a template. When attempting to export a template, you may see an error that states some resources were not exported. If needed, you can manually define these resources in your template after downloading it.

To view and edit these JSON templates, Visual Studio Code or Visual Studio are recommended. Install visual studio code for Windows, OSX or Linux from here : <https://code.visualstudio.com/>

At home: Resources to continue learning

By participating in today’s event you will have exclusive access to download the presentation, step-by-step setup guidance, and latest online training and demos at [aka.ms/itinnovationresources](http://www.aka.ms/itinnovationresources).

|  |  |
| --- | --- |
| Resources | What you’ll learn |
| Event presentation. | Today’s presentation on “What’s new in Azure Infrastructure: Improving Datacenter Flexibility with Microsoft, Open Source and Other Technologies.” |
| Step-by-step guidance (attendee guide) | This document guides you through today’s instructor-led and individual activities, and provide you with the key resources and information necessary to continue building your skills to be able to implement in your environment. |
| Try Azure | Sign up for a free one-month trial of Azure. |
| Access Azure videos and online training | The most recent Enterprise Mobility, On Premises, In Cloud (Azure) and DevOps news and resources, from Microsoft and in the industry. |
| Blog: Regular IT Guy | Stay up to date on the latest in Azure. |

This document supports a preliminary release of a software product that may be changed substantially prior to final commercial release. This document is provided for informational purposes only and Microsoft makes no warranties, either express or implied, in this document. Information in this document, including URL and other Internet Web site references, is subject to change without notice. The entire risk of the use or the results from the use of this document remains with the user. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Copyright 2014 © Microsoft Corporation. All rights reserved.

Microsoft Active Directory, Azure Active Directory, Azure, Hyper-V, Windows, and Windows Server 2012 are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.