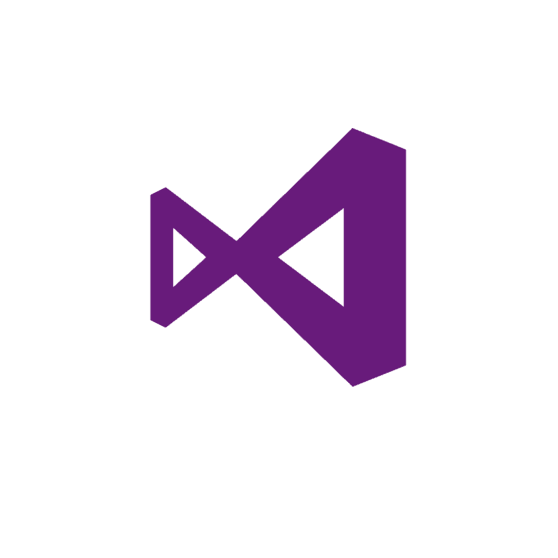
**Hands-On Lab/Demo Script**

**Java Development on Linux with Visual Studio Team Services**

# Exercise 10: Deploying to Azure

Lab version: 1.1.0

Last updated: 3/7/2016



## Overview

In this exercise, you will learn how to deploy the application to Azure using Azure Resource Manager templates to create an Azure Resource Group that will contain a virtual machine installed and configured with Java, Tomcat and MySQL. The resource group will also contain other necessary resources such as storage, network interface card, virtual network, Public IP address, etc.,

There are several starter templates available for ARM. For this exercise, we have taken a basic template and customized to suite the requirements of the application – for instance, configuring MySQL users, creating database and tables, etc.

The customized ARM template is available at <https://github.com/hsachinraj/azure-arm-templates/tree/master/vstsazurejl_arm>.

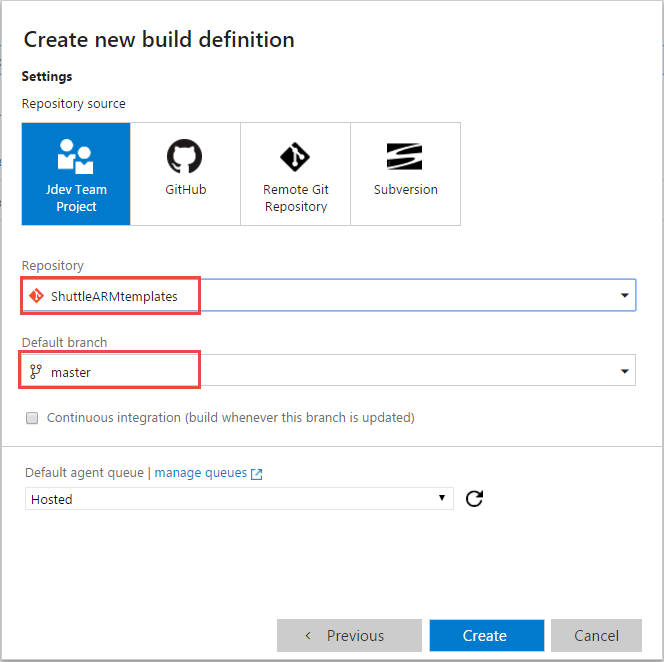
## Pre-requisites for the exercise:

* ARM Templates and the supporting files from the above URL should be added - You can clone the repository to your own GitHub repo or to your VSTS account. We will assume that you have copied the files to your VSTS account repo named **Shuttle ARM templates**.
* Azure supports two different deployment models – ***Classic and Resource Manager***. The lab covers the Resource Manager model which is the recommended model for new deployments. Azure endpoints can be created using *Certificates, Credentials and Service Principal*. While the first two options work for the classic deployment model, Service Principal option is required for Resource Manager model. See this [article](https://blogs.msdn.microsoft.com/visualstudioalm/2015/10/04/automating-azure-resource-group-deployment-using-a-service-principal-in-visual-studio-online-buildrelease-management/) for detailed step-by-step instructions to create an Azure service endpoint using Service Principal in your VSTS account.

Make sure you have completed the above pre-requisites before proceeding further.

## Creating a build definition

Let’s start with a build definition.

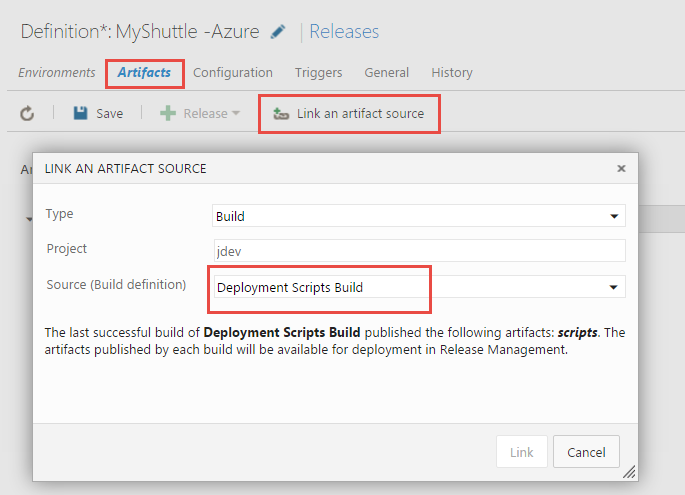
1. Create a new build definition – let’s call it **deployment script build**). Start with an empty template and choose **your team project** as the Repository source. You can use the **Hosted** agent or select an another agent (If you are going to use the same agent for Release, note that agent will need to be on Windows with Azure PowerShell installed). Select **Create** to finish the wizard.  
     
   
2. Add **Copy and publish build artifacts** task to the definition. Set theattributes of the task as following:

|  |  |
| --- | --- |
| **Copy Root** | {leave empty |
| **Contents** | \*.\* |
| **Artifact Name** | Scripts |
| **Artifact Type** | Server |

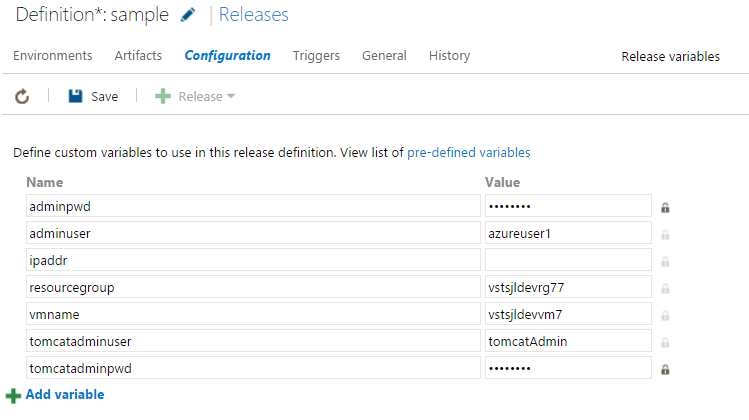
1. Run the build.

## Setting up the Release pipeline

We can now go ahead a create a new Release Definition. Start a new empty release definition – Let’s name it **MyShuttle-Azure**.

1. Link the release to the builds by selecting **Link** **an artifact source**  from the **Artifacts** tab  
     
   
2. We will be using a few variables in our tasks – so, add them to the release definition on the **Configuration** page

|  |  |  |
| --- | --- | --- |
| **Variable Name** | **Value** | **Purpose** |
| adminuser | <<any name>> | name of the admin user for the virtual machine |
| adminpwd | <<your own password>> | Admin password for the virtual machine |
| resourcegroup | <<any name>>, e.g., vstsjldevrg77 | Name for the resource group |
| resourcegroup | <<any name>>, e.g., vstsjldevvm77 | Name for the VM |
| tomcatadminuser | <<any name>> e.g, tomcatAdmin | User that will be added to tomcat users list with manager and admin permissions |
| ipaddr | leave this empty | variable that will be updated at runtime to store the ip address of the VM |

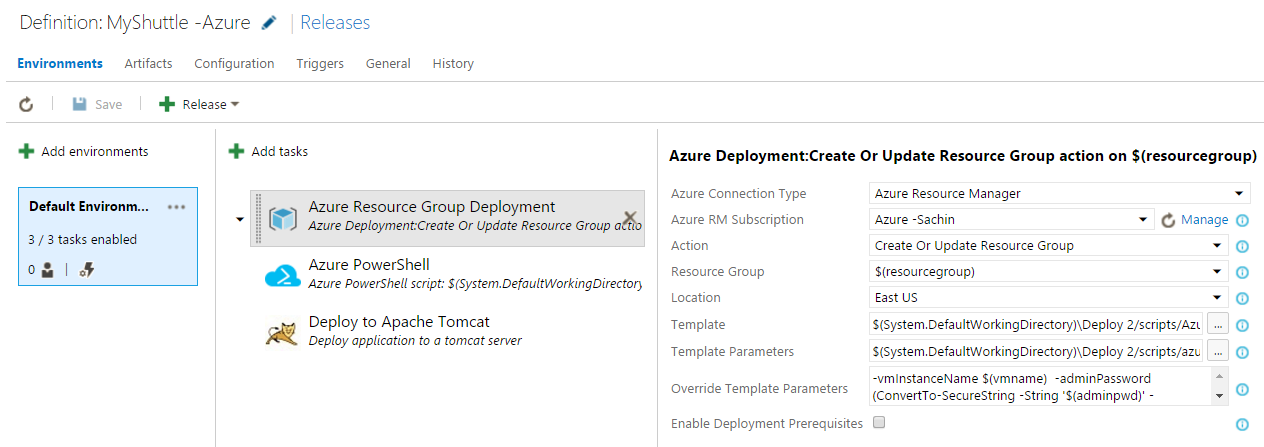


Now, add the **Azure Resource Group Deployment** task to the default environment.

1. Set the task attributes as follows:

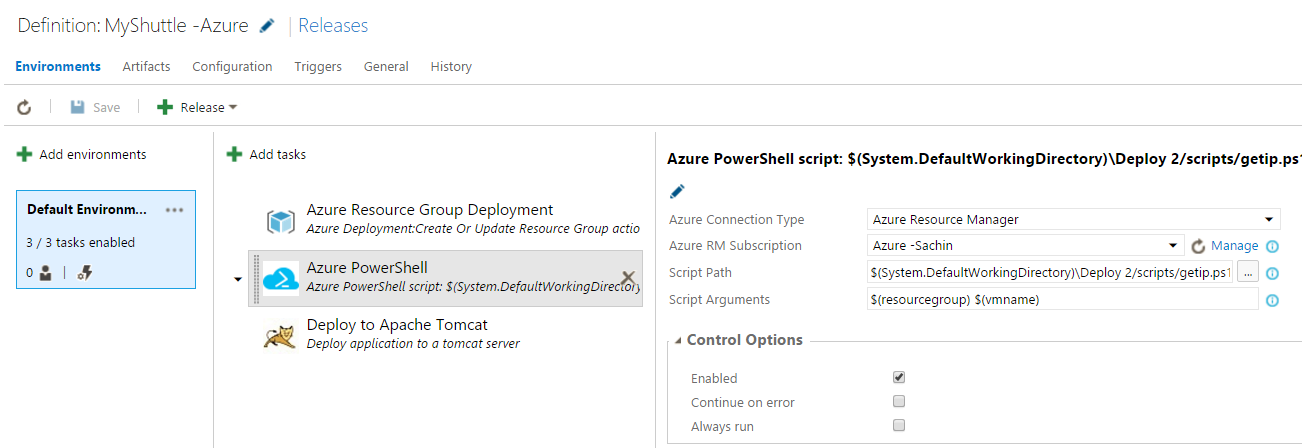
|  |  |
| --- | --- |
| **Attribute** | **Value** |
| Azure Connection Type | Azure Resource Manager |
| Azure RM Subscription | Name of the Azure endpoint you created as described in the [pre-requisite section](#_Pre-requisites_for_the) above |
| Action | Create or Update Resource Group |
| Resource Group | $(resourcegroup) |
| Location | <<any location>> |
| Template | Use the file picker to choose the **azuredeploy.json** file from the build artifacts*. $(System.DefaultWorkingDirectory)\Deploy scripts/scripts/azuredeploy.json* |
| Template Parameters | Use the file picker to choose the **azuredeploy.parameters.json** file from the build artifacts  *$(System.DefaultWorkingDirectory)\Deploy scripts/scripts/azuredeploy.parameters.json* |
| Override Template Parameters | -vmInstanceName $(vmname) -adminUserName $(adminuser) -adminPassword (ConvertTo-SecureString -String '$(adminpwd)' -AsPlainText -Force) -tomcatadminuser $(tomcatadminuser) -tomcatadminpwd $(ConvertTo-SecureString -String '$(adminpwd)' -AsPlainText -Force) |

Here is a screenshot of the task:



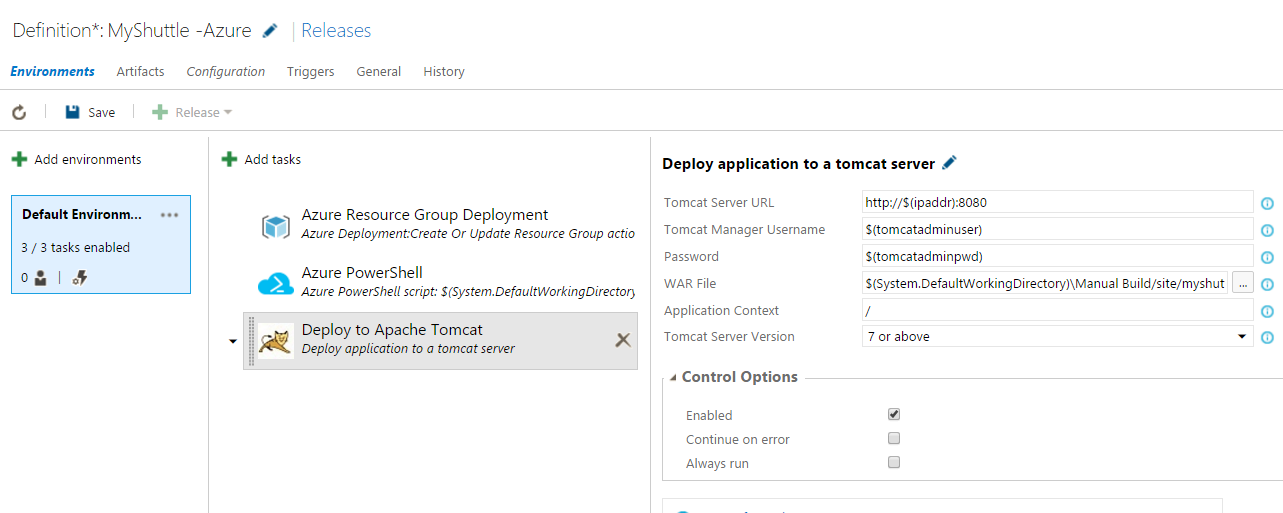
1. Next, add **Azure PowerShell** task. The task will get execute a PowerShell script to retrieve the IP address of the VM created in the resource group and update the ***ipadddr*** variable defined in the RM defintion. Set the attribute of the tasks as follows:

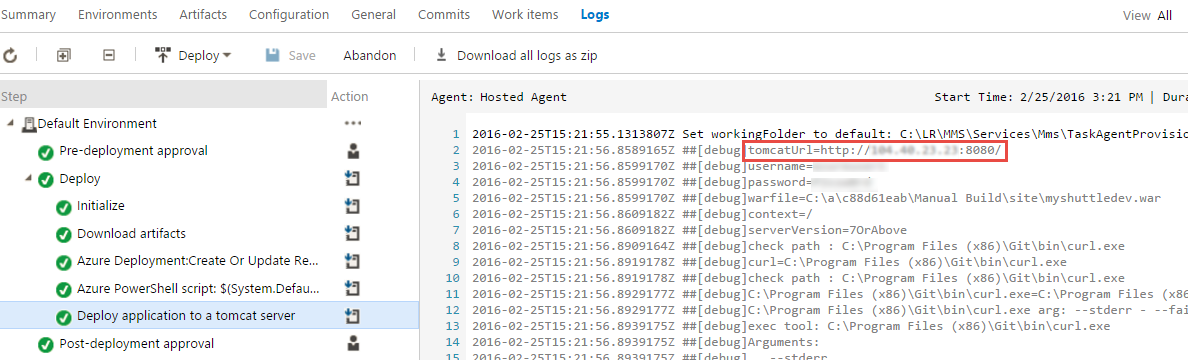
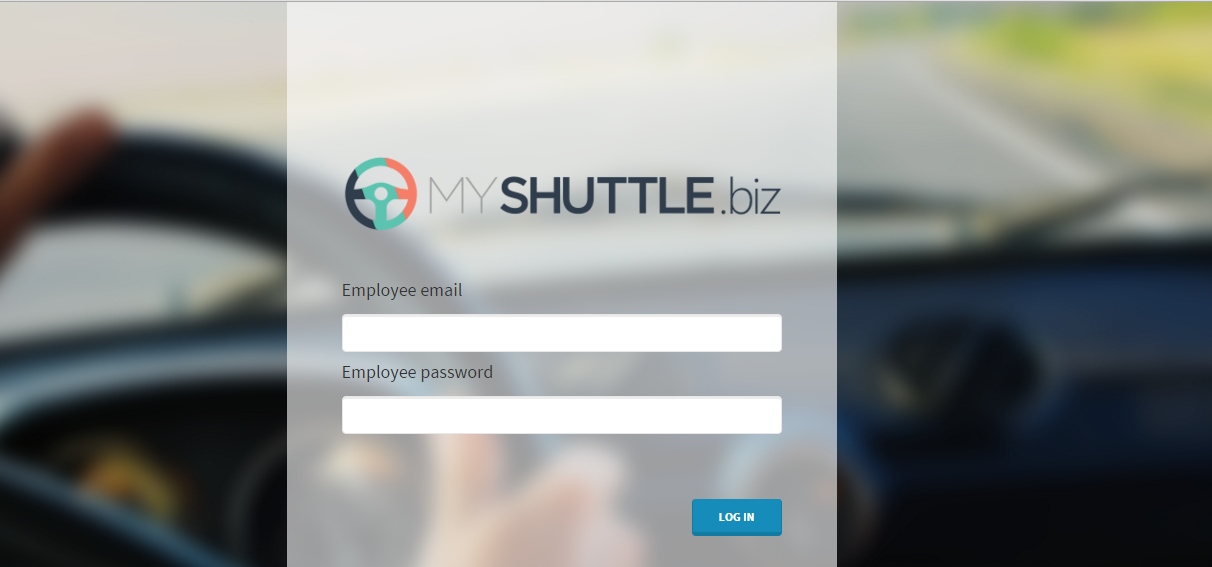
|  |  |
| --- | --- |
| **Attribute** | **Value** |
| Azure Connection Type | Azure Resource Manager |
| Azure RM Subscription | Name of the Azure endpoint you created as described in the [pre-requisite section](#_Pre-requisites_for_the) above |
| Script Path | Use the file picker to choose the **getip.ps1** file from the build artifacts*. $(System.DefaultWorkingDirectory)\Deploy 2/scripts/getip.ps1* |
| Script Arguments | *$(resourcegroup) $(vmname)* |



1. Lastly, add the Tomcat tasks and set the task attributes as specified below:

|  |  |
| --- | --- |
| **Attribute** | **Value** |
| Tomcat Server URL | http://$(ipaddr):8080 |
| Tomcat Manager Username | $(tomcatadminuser) |
| Password | $(tomcatadminpwd) |
| WAR file | Use the file picker to choose the **myshuttledev.war** file from the build artifacts*. - $(System.DefaultWorkingDirectory)\Manual Build/site/myshuttledev.war* |
| Application Context | */* |
| Tomcat Server Version | *7 or Above* |



1. Now, we are ready to run the release. Select **+ Release**  and specify the latest build to start the deployment. Take a coffee break!!- as the provisioning of the resources will take some time. But when it is complete – you will see a VM created and installed with:
   1. Open JDK 7
   2. Tomcat Installed
   3. Tomcat user XML file updated with the role and Admin credentials
   4. MySQL installed
   5. Sample data injected to the database
2. From the log file of the **Apache Tomcat Task**, you can find the URL of the Tomcat server on the VM. 
3. Navigate to the URL and you should see your app deployed!!!  
     
   

To give feedback please write to [devopsdemos@microsoft.com](mailto:devopsdemos@microsoft.com)

Copyright © 2016 by Microsoft Corporation. All rights reserved.