# How to programmatically control Azure Virtual Machine

## Introduction

## To operate Windows Azure virtual machine, using the Azure PowerShell isn't the only way. We also can use management service APIs to achieve this target.

This sample will demonstrate how to create a virtual machine, get its information and delete it.

## Running the Sample

Before running the sample, you must change the variables below to yours.

## -VirtualMachineName

## A unique Name of your VirtualMachine.

## -SettingsFilePath

## Download the publishsettings file from:

<https://manage.windowsazure.com/publishsettings/index?client=vs&schemaversion=2.0&whr=azure.com>

Then set the SettingsFilePath to publishsettings file’s path.

## -SubscriptionID

Set to your Azure subscription ID.

## -StorageAccountName

Set to your storage account name. We need blob storage to store the VM VHD file.

## Using the Code

The sample implements three major Azure Virtual Machine operations.

1. Create Azure Virtual Machine

|  |
| --- |
| -Code block start-  --C# code snippet start--  #region Create Azure Virtual Machine  public static void CreateAzureVM(string vmName, string location = null, string affinityGroup = null)  {  //Step 1:Create Hosted Service  var createHostedServiceResponse=createHostedService(vmName,location,affinityGroup);  //A successful operation returns status code 201 (Created).  if ((int)createHostedServiceResponse.StatusCode == 201)  {  Console.WriteLine("Create cloud service success!");  //Step 2: Create Virtual Machin Deployment  var createDeploymentResponse = createVMDeployment(vmName, vmName, vmName);  if ((int)createDeploymentResponse.StatusCode==202)  {  Console.WriteLine("Create VM success!");  }  else  {  Console.WriteLine("Error:" + getErrorMessageFromResponse(createDeploymentResponse));  }  }  else  {  Console.WriteLine("Error:"+getErrorMessageFromResponse(createHostedServiceResponse));  }  }  private static HttpWebResponse createHostedService(string serviceName, string location = null, string affinityGroup = null)  {  string locationOrAffinity = string.Empty;  string value = string.Empty;  if (string.IsNullOrEmpty(location) == false)  {  locationOrAffinity = "Location";  value = location;  }  else  {  locationOrAffinity = "AffinityGroup";  value = affinityGroup;  }  XDocument requestBody = new XDocument(  new XElement(ns + "CreateHostedService",  new XElement(ns + "ServiceName", serviceName),  new XElement(ns + "Label", convertToBase64(serviceName)),  new XElement(ns + locationOrAffinity, value))  );  var response = sendHttpReqeust(  "<https://management.core.windows.net/>" + SubscriptionID + "/services/hostedservices",  "POST",  ClientCertificate,  ContentType,  Version,  requestBody  );  return response;  }  private static HttpWebResponse createVMDeployment(string cloudServiceName, string deploymentName, string VMName)  {  var now = DateTime.UtcNow;  string dateString = now.Year + "-" + now.Month + "-" + now.Day + now.Hour + now.Minute + now.Second + now.Millisecond;  XDocument requestBody = new XDocument(  new XElement(ns + "Deployment",  new XElement(ns + "Name", deploymentName),  new XElement(ns + "DeploymentSlot", DeploymentSlot),  new XElement(ns + "Label", convertToBase64(deploymentName)),  new XElement(ns + "RoleList",  new XElement(ns + "Role",  new XElement(ns + "RoleName", VMName),  new XElement(ns + "RoleType", "PersistentVMRole"),  new XElement(ns + "ConfigurationSets",  new XElement(ns + "ConfigurationSet",  new XElement(ns + "ConfigurationSetType", "WindowsProvisioningConfiguration"),  new XElement(ns + "InputEndpoints",  new XElement(ns + "InputEndpoint",  new XElement(ns + "LocalPort", 3389),  new XElement(ns + "Name", "RDP"),  new XElement(ns + "Protocol", "tcp")),  new XElement(ns + "InputEndpoint",  new XElement(ns + "LocalPort", 80),  new XElement(ns + "Name", "web"),  new XElement(ns + "Port", 80),  new XElement(ns + "Protocol", "tcp"))),  new XElement(ns + "ComputerName", VMName),  new XElement(ns + "AdminPassword", AdminPassword),  new XElement(ns + "AdminUsername", AdminUsername))),  new XElement(ns + "Label", VMName),  new XElement(ns + "OSVirtualHardDisk",  new XElement(ns + "MediaLink", string.Format("http://{0}.blob.core.windows.net/vhds/{1}.vhd", StorageAccountName, dateString)),  new XElement(ns + "SourceImageName", getSourceImageNameByFamliyName(ImageFamilyName))),  new XElement(ns + "RoleSize", RoleSize)  ))));  var response = sendHttpReqeust(  string.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments", SubscriptionID, cloudServiceName),  "POST",  ClientCertificate,  ContentType,  Version,  requestBody);  return response;  }  private static string convertToBase64(string targetString)  {  System.Text.ASCIIEncoding ae = new System.Text.ASCIIEncoding();  byte[] svcNameBytes = ae.GetBytes(targetString);  return Convert.ToBase64String(svcNameBytes);  }  private static string getSourceImageNameByFamliyName(string imageFamliyName)  {  string imageName = null;  var response = sendHttpReqeust(  string.Format("https://management.core.windows.net/{0}/services/images", SubscriptionID),  "GET",  ClientCertificate,  "application/xml",  "2014-02-01",  null  );  using (Stream responseStream = response.GetResponseStream())  {  using (StreamReader reader = new StreamReader(responseStream))  {  var imagesXML = reader.ReadToEnd();  imageName = XElement.Parse(imagesXML).Elements()  .Where(o => o.Descendants(ns + "ImageFamily").Count() > 0)  .Where(o => o.Element(ns + "ImageFamily").Value.ToString() == imageFamliyName)  .Last().Element(ns + "Name")  .Value.ToString();  }  }  return imageName;  }  #endregion  --C# code snippet end--  --VB code snippet start--  #Region "Create Azure Virtual Machine"  Public Sub CreateAzureVM(vmName As String, Optional location As String = Nothing, Optional affinityGroup As String = Nothing)  'Step 1:Create Hosted Service  Dim createHostedServiceResponse = createHostedService(vmName, location, affinityGroup)  'A successful operation returns status code 201 (Created).  If CInt(createHostedServiceResponse.StatusCode) = 201 Then  Console.WriteLine("Create cloud service success!")  'Step 2: Create Virtual Machin Deployment  Dim createDeploymentResponse = createVMDeployment(vmName, vmName, vmName)  If CInt(createDeploymentResponse.StatusCode) = 202 Then  Console.WriteLine("Create VM success!")  Else  Console.WriteLine("Error:" & getErrorMessageFromResponse(createDeploymentResponse))  End If  Else  Console.WriteLine("Error:" & getErrorMessageFromResponse(createHostedServiceResponse))  End If  End Sub  Private Function createHostedService(serviceName As String, Optional location As String = Nothing, Optional affinityGroup As String = Nothing) As HttpWebResponse  Dim locationOrAffinity As String = String.Empty  Dim value As String = String.Empty  If String.IsNullOrEmpty(location) = False Then  locationOrAffinity = "Location"  value = location  Else  locationOrAffinity = "AffinityGroup"  value = affinityGroup  End If  Dim requestBody As New XDocument(New XElement(ns.ToString() & "CreateHostedService", New XElement(ns.ToString() & "ServiceName", serviceName), New XElement(ns.ToString() & "Label", convertToBase64(serviceName)), New XElement(ns.ToString() & locationOrAffinity, value)))  Dim response = sendHttpReqeust("https://management.core.windows.net/" & SubscriptionID & "/services/hostedservices", "POST", ClientCertificate, ContentType, Version, requestBody)  Return response  End Function  Private Function createVMDeployment(cloudServiceName As String, deploymentName As String, VMName As String) As HttpWebResponse  Dim now = DateTime.UtcNow  Dim dateString As String = Convert.ToString(now.Year) & "-" & Convert.ToString(now.Month) & "-" & Convert.ToString(now.Day) & Convert.ToString(now.Hour) & Convert.ToString(now.Minute) & Convert.ToString(now.Second) & Convert.ToString(now.Millisecond)  Dim requestBody As New XDocument(New XElement(ns.ToString() & "Deployment", New XElement(ns.ToString() & "Name", deploymentName), New XElement(ns.ToString() & "DeploymentSlot", DeploymentSlot), New XElement(ns.ToString() & "Label", convertToBase64(deploymentName)), New XElement(ns.ToString() & "RoleList", New XElement(ns.ToString() & "Role", New XElement(ns.ToString() & "RoleName", VMName), New XElement(ns.ToString() & "RoleType", "PersistentVMRole"), New XElement(ns.ToString() & "ConfigurationSets", New XElement(ns.ToString() & "ConfigurationSet", New XElement(ns.ToString() & "ConfigurationSetType", "WindowsProvisioningConfiguration"), New XElement(ns.ToString() & "InputEndpoints", New XElement(ns.ToString() & "InputEndpoint", New XElement(ns.ToString() & "LocalPort", 3389), New XElement(ns.ToString() & "Name", "RDP"), New XElement(ns.ToString() & "Protocol", "tcp")), New XElement(ns.ToString() & "InputEndpoint", New XElement(ns.ToString() & "LocalPort", 80), New XElement(ns.ToString() & "Name", "web"), New XElement(ns.ToString() & "Port", 80), New XElement(ns.ToString() & "Protocol", "tcp"))), New XElement(ns.ToString() & "ComputerName", VMName), New XElement(ns.ToString() & "AdminPassword", AdminPassword), New XElement(ns.ToString() & "AdminUsername", AdminUsername))), New XElement(ns.ToString() & "Label", VMName), New XElement(ns.ToString() & "OSVirtualHardDisk", New XElement(ns.ToString() & "MediaLink", String.Format("http://{0}.blob.core.windows.net/vhds/{1}.vhd", StorageAccountName, dateString)), New XElement(ns.ToString() & "SourceImageName", getSourceImageNameByFamliyName(ImageFamilyName))), \_  New XElement(ns.ToString() & "RoleSize", RoleSize)))))  Dim response = sendHttpReqeust(String.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments", SubscriptionID, cloudServiceName), "POST", ClientCertificate, ContentType, Version, requestBody)  Return response  End Function  Private Function convertToBase64(targetString As String) As String  Dim ae As New System.Text.ASCIIEncoding()  Dim svcNameBytes As Byte() = ae.GetBytes(targetString)  Return Convert.ToBase64String(svcNameBytes)  End Function  Private Function getSourceImageNameByFamliyName(imageFamliyName As String) As String  Dim imageName As String = Nothing  Dim response = sendHttpReqeust(String.Format("https://management.core.windows.net/{0}/services/images", SubscriptionID), "GET", ClientCertificate, "application/xml", "2014-02-01", Nothing)  Using responseStream As Stream = response.GetResponseStream()  Using reader As New StreamReader(responseStream)  Dim imagesXML = reader.ReadToEnd()  imageName = XElement.Parse(imagesXML).Elements().Where(Function(o) o.Descendants(ns.ToString() & "ImageFamily").Count() > 0).Where(Function(o) o.Element(ns.ToString() & "ImageFamily").Value.ToString() = imageFamliyName).Last().Element(ns.ToString() & "Name").Value.ToString()  End Using  End Using  Return imageName  End Function  #End Region  --VB code snippet end--  -Code block end- |

1. Get Azure Virtual Machine information

|  |
| --- |
| -Code block start-  --C# code snippet start--  --VB code snippet start--  #region Get Azure Vitrual Machine Informations  public static void GetAzureVMInformations(string vmName)  {  Console.WriteLine("Start to get VM {0}'s informations",vmName);  var response = getAzureVM(vmName, vmName, vmName);  if ((int)response.StatusCode==200)  {  using (Stream responseStream=response.GetResponseStream())  {  using (StreamReader reader = new StreamReader(responseStream))  {  var imagesXML = reader.ReadToEnd();  Console.WriteLine(imagesXML);  }  }  }  else  {  Console.WriteLine("Error:" + getErrorMessageFromResponse(response));  }  }  private static HttpWebResponse getAzureVM(string servicName, string deploymentName, string vmName)  {  var response = sendHttpReqeust(string.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments/{2}/roles/{3}", SubscriptionID, servicName, deploymentName, vmName),  "GET",  ClientCertificate,  ContentType,  Version,  null);  return response;  }  #endregion  --VB code snippet end--  --C# code snippet end--  --VB code snippet start--  #Region "Get Azure Vitrual Machine Informations"  Public Sub GetAzureVMInformations(vmName As String)  Console.WriteLine("Start to get VM {0}'s informations", vmName)  Dim response = getAzureVM(vmName, vmName, vmName)  If CInt(response.StatusCode) = 200 Then  Using responseStream As Stream = response.GetResponseStream()  Using reader As New StreamReader(responseStream)  Dim imagesXML = reader.ReadToEnd()  Console.WriteLine(imagesXML)  End Using  End Using  Else  Console.WriteLine("Error:" & getErrorMessageFromResponse(response))  End If  End Sub  Private Function getAzureVM(servicName As String, deploymentName As String, vmName As String) As HttpWebResponse  Dim response = sendHttpReqeust(String.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments/{2}/roles/{3}", SubscriptionID, servicName, deploymentName, vmName), "GET", ClientCertificate, ContentType, Version, Nothing)  Return response  End Function  #End Region  --VB code snippet end--  -Code block end- |

1. Delete Azure Virtual Machine(This method only works when you have multiple VMs in VM deployment)

|  |
| --- |
| -Code block start-  --C# code snippet start--  #region Delete Azure Virtual Machine  public static void DeleteAzureVM(string vmName)  {  Console.WriteLine("Start to delete Azure VM, VM Name={0} this delete operation only work when there are multiple Virtual Machines in one VMDeployment",vmName);  var response = deleteVM(vmName, vmName, vmName);  if ((int)response.StatusCode==202)  {  Console.WriteLine("Delete success!");  }  else  {  Console.WriteLine("Error:" + getErrorMessageFromResponse(response));  }  }  private static HttpWebResponse deleteVM(string servicName, string deploymentName, string vmName)  {  var response = sendHttpReqeust(string.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments/{2}/roles/{3}?comp=media", SubscriptionID, servicName, deploymentName, vmName),  "DELETE",  ClientCertificate,  null,  Version,  null);  return response;  }  #endregion  --C# code snippet end--  --VB code snippet start--  #Region "Delete Azure Virtual Machine"  Public Sub DeleteAzureVM(vmName As String)  Console.WriteLine("Start to delete Azure VM, VM Name={0} this delete operation only work when there are multiple Virtual Machines in one VMDeployment", vmName)  Dim response = deleteVM(vmName, vmName, vmName)  If CInt(response.StatusCode) = 202 Then  Console.WriteLine("Delete success!")  Else  Console.WriteLine("Error:" & getErrorMessageFromResponse(response))  End If  End Sub  Private Function deleteVM(servicName As String, deploymentName As String, vmName As String) As HttpWebResponse  Dim response = sendHttpReqeust(String.Format("https://management.core.windows.net/{0}/services/hostedservices/{1}/deployments/{2}/roles/{3}?comp=media", SubscriptionID, servicName, deploymentName, vmName), "DELETE", ClientCertificate, Nothing, Version, Nothing)  Return response  End Function  #End Region  --VB code snippet end--  -Code block end- |

## More Information

msdn.microsoft.com/en-us/library/azure/ee460799.aspx