Demo Script: Azure Container Registry -Getting Started

Steve Lasker

[Steve.Lasker@Microsoft.com](mailto:Steve.Lasker@Microsoft.com)

<http://blogs.msdn.com/SteveLasker>

# Demo environment setup & prereqs

## Install and Configure Docker For Windows Beta

[Setting Up Docker For Windows (D4W)](onenote:Piñata.one#Setting%20Up%20Docker%20For%20Windows%20(D4W)&section-id={3A79CD1D-D9A5-484C-85FE-7B734C48A4F4}&page-id={43D2985A-D59C-455D-A413-69A585BBC0C8}&end&base-path=https://microsoft.sharepoint.com/teams/CPT/AzureTools/Shared%20Documents/Docker/Docker%20Investiga)

[Azure CLI](https://github.com/Azure/azure-cli)

# Cached Docker Images for ASP.NET

Run the following in a PowerShell prompt

# Demo Reset

**docker rm -f $(docker ps -a -q)**

## Cache Images

**docker pull microsoft/aspnetcore:1.0.1**

**docker pull microsoft/aspnetcore-build:1.0.1**

**docker pull stevelasker/helloworld**

# Create New Registry – Azure Portal

|  |  |  |
| --- | --- | --- |
|  | **Demo step** | **Talk track & notes** |
|  | **Search for Container Registry** | We’ll find the resource type of Container Registry |
|  |  | Here we have the Azure Container Registries as they exist in the Azure Portal |
|  | **Click Add** | We can easily create a new registry here in the portal |
|  | **Registry Name: AzureDevEx**  **Subscription: \_\_\_**  **Resource Group: [use an existing to speed things up]**  **Location: [set based on the resource group location]**  **Admin user: Enabled** | We can set a basic set of values for one of the active subscriptions you may have  We can use an existing resource group which will set the location  The interesting thing here is the Admin account. This is a single user account that you can start with.  You’ll likely want to use Service Principals for headless services, like your Build system, or your Container deployment that needs access to pull images  While we have Service Principal integration in the Preview, individual and group identities will come as we get closer to our GA/General Availably release in the spring. |
|  | **Browse BikeSharingTest** | While that registry is being created, we can look at another one already configured. |
|  | **Click on Overview**  **Point to Login Server** | We can see our registry name has a suffix of our primary domain for our subscription. For me, that’s Microsoft. |
|  | **Select Access Keys** | Looking at the Access Keys we can see the Admin Account details |
|  | **Select Access control** | Here we can see the users and service principals that have access to the registry.  These individual users have access to configure the registry, but these aren’t users to access the registry.  For registry access, you’ll need the admin account or the username and password from a service principal |
|  | **Launch PowerShell** | Now, you may want a CLI to configure the Azure Container Registry.  Here I can use the Azure v2 CLI, which is an open source CLI. |
|  | **Docker login AzureDevEx-microsoft.azurecr.io -u [usename] – p [password]** | To access our new registry, we can use the standard docker CLI for login, push, pull |
|  | **docker tag helloworld AzureDevEx-microsoft.azurecr.io/helloworld** | We’ll tag our helloworld image with our registry login URL |
|  | **docker push AzureDevEx-microsoft.azurecr.io/helloworld** | and push it |
|  | **docker tag helloworld AzureDevEx-microsoft.azurecr.io/marketing/2016campaign/web** | The Azure container registry also supports multiple nested namespaces. This allows you to group a collection of common images within a specific registry. This avoids having to create multiple registries for each collection a specific group has.  You can think of this as  [org]-[company].azurecr.io/[app-grouping]/[sub-group]/[image]:[tag] |
|  | **docker push AzureDevEx-microsoft.azurecr.io/marketing/2016campaign/web** | We can push this additional one to the registry |
|  | **aka.ms/acr/manage**  **login: azuredevex-microsoft.azurecr.io**  **user**  **password** | Once your registry starts collecting images, you may want to view the contents of the registry. You may have forgotten the exact image name, or the tags |
|  | **Click through the images** |  |
|  | **Switch to bikesharing-ex.azurecr.io** |  |

# Using the Azure CLI

|  |  |  |
| --- | --- | --- |
|  | **Demo step** | **Talk track & notes** |
|  | **az** | In addition to the docker cli for interacting with the Azure Container Registry, we can also use the Open Source Azure CLI for configuring the Azure Container Registry |
|  | **az acr create -l southcentralus -n stevelas3 -g deleteme --admin-enabled true** | We can create a registry here using the Azure v2 Cross Platform CLI  This CLI is available on Windows, Mac and Linux |
|  | **az ad sp create-for-rbac --scopes /subscriptions/daae1e1a-63dc-454f-825d-b39289070f79/resourcegroups/deleteme/providers/Microsoft.ContainerRegistry/registries/stevelas3 --role Owner --password passw0rd** | Once we’ve created a registry, we have the option to associate Azure AD Service Principals. Using Service Principals we can configure build systems like VSTS |
|  | **az acr credential show -n stevelas3** | We can also see the admin credentials |
|  | **az acr repository list -n stevelas3**  **az acr repository list -n dockerdemos** | And, we can use the docker catalog api to see the list of repositories (collections of images) in our registry |
|  | **az acr repository show-tags -n dockerdemos --repository bikesharing.campaign** | And, the tags for each repository |

# Quick List

|  |
| --- |
| **Display Registries in Portal** |
| **Add SteveLas3** |
| **Browse SteveLas3 in Portal** |
| **Browse DockerDemos in portal** |
| **Click Overview**  **Login URL**  **Access Keys**  **Access control (IAM)** |
| **Az acr list** |
| **az acr create -n stevelas3**  **-l westus**  **-g deleteme**  **--admin-enabled true** |
| **Az ad sp create…** |
| **Copy username/password** |
| **PowerShell**  **Docker login**  **Docker tag helloworld …**  **Docker push** |
| **Nested namespaces**  **Docker tag marketing/campaign**  **Docker push** |
| **Browse Portal** |
| **App Service** |
|  |
|  |