Use a Cognitive Services Container

Using cognitive services hosted in Azure enables application developers to focus on the infrastructure for their own code while benefiting from scalable services that are managed by Microsoft. However, in many scenarios, organizations require more control over their service infrastructure and the data that is passed between services.

Many of the cognitive services APIs can be packaged and deployed in a *container*, enabling organizations to host cognitive services in their own infrastructure; for example in local Docker servers, Azure Container Instances, or Azure Kubernetes Services clusters. Containerized cognitive services need to communicate with an Azure-based cognitive services account to support billing; but application data is not passed to the back-end service, and organizations have greater control over the deployment configuration of their containers, enabling custom solutions for authentication, scalability, and other considerations.

Clone the repository for this course

If you have already cloned **AI-102-AIEngineer** code repository to the environment where you're working on this lab, open it in Visual Studio Code; otherwise, follow these steps to clone it now.

1. Start Visual Studio Code.
2. Open the palette (SHIFT+CTRL+P) and run a **Git: Clone** command to clone the `https://github.com/MicrosoftLearning/AI-102-AIEngineer` repository to a local folder (it doesn't matter which folder).
3. When the repository has been cloned, open the folder in Visual Studio Code.
4. Wait while additional files are installed to support the C# code projects in the repo.

> **Note**: If you are prompted to add required assets to build and debug, select **Not Now**.

Provision a Cognitive Services resource

If you don't already have one in your subscription, you'll need to provision a **Cognitive Services** resource.

1. Open the Azure portal at `https://portal.azure.com`, and sign in using the Microsoft account associated with your Azure subscription.
2. Select the **+ Create a resource** button, search for *cognitive services*, and create a **Cognitive Services** resource with the following settings:
    - **Subscription**: *Your Azure subscription*
    - **Resource group**: *Choose or create a resource group (if you are using a restricted subscription, you may not have permission to create a new resource group - use the one provided)*
    - **Region**: *Choose any available region*
    - **Name**: *Enter a unique name*
    - **Pricing tier**: Standard S0
3. Select the required checkboxes and create the resource.
4. Wait for deployment to complete, and then view the deployment details.
5. When the resource has been deployed, go to it and view its **Keys and Endpoint** page. You will need the endpoint and one of the keys from this page in the next procedure.

Deploy and run a Text Analytics container

Many commonly used cognitive services APIs are available in container images. For a full list, check out the cognitive services documentation. In this exercise, you'll use the container image for the Text Analytics *language detection* API; but the principles are the same for all of the available images.

1. In the Azure portal, on the **Home** page, select the **+ Create a resource** button, search for *container instances*, and create a **Container Instances** resource with the following settings:
    - **Basics**:

- **Subscription**: *Your Azure subscription*
- **Resource group**: *Choose the resource group containing your cognitive services resource*
- **Container name**: *Enter a unique name*
- **Region**: *Choose any available region*
- **Image source**: Docker Hub or other Registry
- **Image type**: Public
- **Image**:
  `mcr.microsoft.com/azure-cognitive-services/textanalytics/language:1.1.013570001-amd64`
- **OS type**: Linux
- **Size**: 1 vcpu, 4 GB memory

- **Networking**:
  - **Networking type**: Public
  - **DNS name label**: *Enter a unique name for the container endpoint*
  - **Ports**: *Change the TCP port from 80 to* ***5000***
- **Advanced**:
  - **Restart policy**: On failure
  - **Environment variables**:

Mark as secure	Key	Value
Yes	`ApiKey`	*Either key for your cognitive services resource*
Yes	`Billing`	*The endpoint URI for your cognitive services resource*
No	`Eula`	`accept`

  - **Command override**: [ ]
- **Tags**:
  - *Don't add any tags*

2. Wait for deployment to complete, and then go to the deployed resource.
3. Observe the following properties of your container instance resource on its **Overview** page:
   - **Status**: This should be *Running*.
   - **IP Address**: This is the public IP address you can use to access your container instances.
   - **FQDN**: This is the *fully-qualified domain name* of the container instances resource, you can use this to access the container instances instead of the IP address.

> **Note**: In this exercise, you've deployed the cognitive services container image for text translation to an Azure Container Instances (ACI) resource. You can use a similar approach to deploy it to a *Docker* host on your own computer or network by running the following command (on a single line) to deploy the language detection container to your local Docker instance, replacing *<yourEndpoint>* and *<yourKey>* with your endpoint URI and either of the keys for your cognitive services resource.
>
> Code | Copy
>
> ```
> docker run --rm -it -p 5000:5000 --memory 4g --cpus 1 mcr.microsoft.c cognitive-services/textan Eula=accept Billing= <yourEndpoint> ApiKey= <yourKey>
> ```
>
> The command will look for the image on your local machine, and if it doesn't find it there it will pull it from the *mcr.microsoft.com* image registry and deploy it to your Docker instance. When deployment is complete, the container will start and listen for incoming requests on port 5000.

Use the container

1. In Visual Studio Code, in the **04-containers** folder, open **rest-test.cmd** and edit the **curl** command it contains (shown below), replacing *<your_ACI_IP_address_or_FQDN>* with the IP address or FQDN for your container.

```
curl -X POST "http://<your_ACI_IP_address_or_FQDN>:5000/text/analytics/v3.0/languages?" -H "Content-Type: application/json" --data-ascii "{'documents':[{'id':1,'text':'Hello world.'},{'id':2,'text':'Salut tout le monde.'}]}"
```

2. Save your changes to the script. Note that you do not need to specify the cognitive services endpoint or key - the request is processed by the containerized service. The container in turn communicates periodically with the service in Azure to report usage for billing, but does not send request data.
3. Right-click the **04-containers** folder and open an integrated terminal. Then enter the following command to run the script:

```
rest-test
```

4. Verify that the command returns a JSON document containing information about the language detected in the two input documents (which should be English and French).

Clean Up

If you've finished experimenting with your container instance, you should delete it.

1. In the Azure portal, open the resource group where you created your resources for this exercise.
2. Select the container instance resource and delete it.

More information

For more information about containerizing cognitive services, see the [Cognitive Services containers documentation].