



Processor



Deployment



PROCESSOR > GUIDES > DEPLOYMENT

Deploying to Microsoft Azure



PSPDFKit Processor has been deprecated and replaced by [Document Engine](#). To migrate to Document Engine and unlock advanced document processing capabilities, refer to our migration guide. Learn more about these enhancements on our [blog](#).

This guide will walk you through the steps for deploying PSPDFKit Processor to the Microsoft Azure Kubernetes Service with [Kubernetes](#).

Setting Up Azure CLI

To deploy PSPDFKit Processor to the [Microsoft Azure Kubernetes Service](#) with [Kubernetes](#), you have to set up the [Azure CLI](#) utility to manage your [Kubernetes](#) cluster in the command line.

To install [Azure CLI](#), follow the installation instructions from the [Azure CLI installation guide](#).

After you've installed [Azure CLI](#), run the following command to log in to [Microsoft Azure](#):

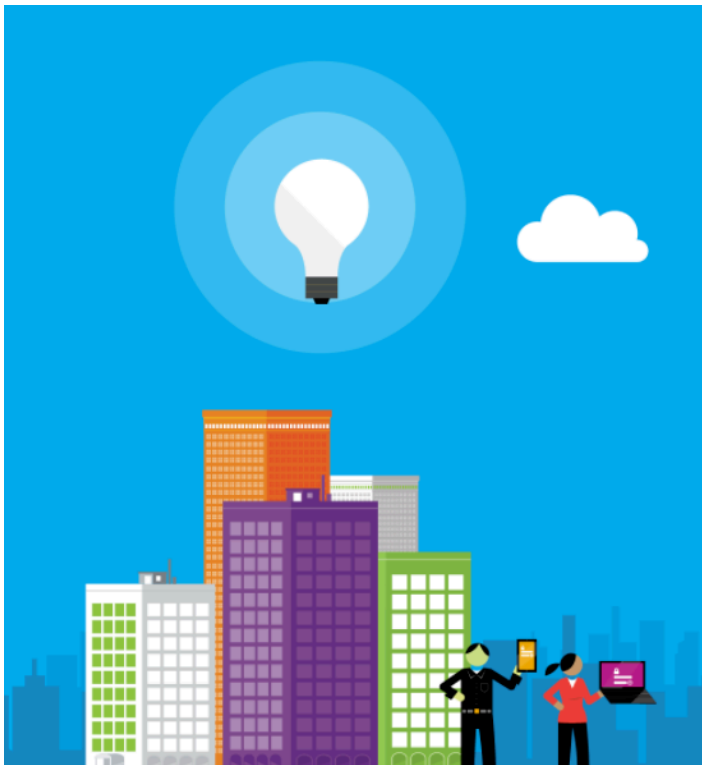
```
az login
```



This command will print the URL <https://microsoft.com/devicelogin> and the code for signing in. Open the URL in your browser and enter the code to sign in to your [Microsoft Azure](#) account.



ASK AI



Device Login

Enter the code that you received from the application on your device

©2018 Microsoft
[Terms of use](#) [Privacy & cookies](#)



Creating a Resource Group

To create a resource group, run the following:

```
az group create -l eastus -n pspdfkitresourcegroup
```



In this example, we created the resource group in the `eastus` region with the name `pspdfkitresourcegroup`. An overview of available regions can be found on Microsoft's Azure geographies page.

Creating a Kubernetes Cluster

To manage your Kubernetes cluster from the command line, you have to install `kubectl`:

```
az aks install-cli
```



To create a Kubernetes cluster with the name `pspdfkitAKScluster`, run the following:

Note: Microsoft Azure trials are limited to four vCPUs. `aks create` needs six vCPUs by default (three nodes × two CPUs of `Standard_D2` VM size). To create a cluster within free account limits, we recommend the following to generate a cluster with two nodes each using the default `Standard_D2` VM size:

```
az aks create -g pspdfkitresourcegroup --name pspdfkitAKScluster --generate-ssh-key
```

To connect `kubectl` with your cluster, execute:

```
az aks get-credentials -g pspdfkitresourcegroup -n pspdfkitAKScluster
```

Creating a ConfigMap

[ConfigMaps](#) allow you to decouple configuration artifacts from image content. To create the `pspdfkit-config` [ConfigMap](#), run the following command:

```
kubectl create configmap pspdfkit-config
```

After the [ConfigMap](#) is created, you can edit it with the following:

```
kubectl edit configmap pspdfkit-config
```

This will open the created [ConfigMap](#) in your editor. Edit the file to match the following file, and replace `activation_key` with your activation key:

```
1  # Please edit the object below. Lines beginning with a '#' will be ignored,
2  # and an empty file will abort the edit. If an error occurs while saving,
3  # this file will be reopened with the relevant failures.
4  #
5  apiVersion: v1
6  data:
```

```
7     api_auth_token: secret
8     license_key: YOUR_LICENSE_KEY_GOES_HERE
9     kind: ConfigMap
```

Don't change anything that comes after the `kind: ConfigMap` line, because that part is autogenerated.

Creating Deployments

To run PSPDFKit Processor, you have to define a Deployment for PSPDFKit Processor. Kubernetes Deployments can be configured in a file. To do so, you'll need to create the configuration for PSPDFKit Processor (`processor.yml`) and ensure that the `pspdfkit/processor` image tag corresponds to the latest PSPDFKit Processor version:

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: processor
5  spec:
6    ports:
7      - protocol: TCP
8        port: 5000
9        targetPort: 5000
10   selector:
11     app: processor
12   type: LoadBalancer
13 ---
14 apiVersion: apps/v1
15 kind: Deployment
16 metadata:
17   name: processor
18 spec:
19   selector:
20     matchLabels:
21       app: processor
22   template:
23     metadata:
24       labels:
25         app: processor
26     spec:
27       containers:
28         - image: "pspdfkit/processor:2023.11.1"
29           name: processor
30           env:
31             - name: ACTIVATION_KEY
32               valueFrom:
33                 configMapKeyRef:
34                   name: pspdfkit-config
35                   key: license_key
```

```
36         - name: API_AUTH_TOKEN
37           valueFrom:
38             configMapKeyRef:
39               name: pspdfkit-config
40               key: api_auth_token
41     ports:
42       - containerPort: 5000
43         name: processor
```

To create the Deployments needed to run PSPDFKit Processor, execute:

```
kubectl create -f ./pspdfkit-processor.yml
```

Accessing the Processor Service

To access this new Processor instance, you have to get the external IP address that was assigned to the service. Run the following command to view all the Services in your cluster, along with their assigned external IP addresses:

```
kubectl get services
```

This will show something like the following:

	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
2	kubernetes	ClusterIP	10.15.240.1	<none>	443/TCP
3	processor	LoadBalancer	10.15.247.197	12.345.678.910	5000:32393/TCP

Copy the `EXTERNAL-IP` address from the `processor` row and use it together with the port `5000` in your web browser. In this example, you'll use `http://12.345.678.910:5000/`. This will present you with a welcome message:

```
PSPDFKit Processor is up and running.
```

You just confirmed that your Processor instance is up and running. You can now post processing requests. For example:

```
1 curl -H "Authorization: Token token=secret" \  
2   -F file=@demo.pdf \  
3   -F operations="{\"operations\": [{\"type\": \"rotatePages\", \"pageIndexes\": \">  
4   http://12.345.678.910:5000/process \  
5   --output result.pdf
```



Limitations

Be aware that this is just an example setup, and we recommend looking deeper into the Microsoft Azure Kubernetes Service for a production-ready setup.

Was this helpful?

 YES

 NO

Questions? [Contact us](#)

