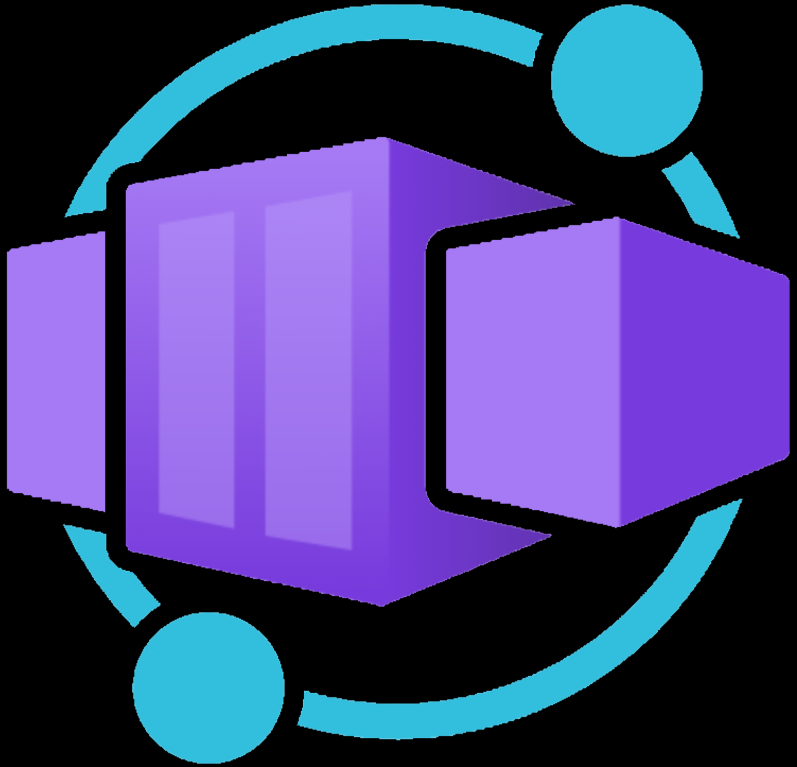
# Container Apps deploy with YAML

```yaml
type: Microsoft.App/containerApps
tags:
  env: production
properties:
  managedEnvironmentId: <env_id>
  configuration:
    activeRevisionsMode: Multiple
    ingress:
      external: true
      allowInsecure: false
      targetPort: 80
  template:
    revisionSuffix: myrevision
    containers:
      - image: nginx
        name: nginx
```



**Houssem Dellai**
Cloud Solution Architect at Microsoft

# Creating Container Apps using Azure CLI - options

```
az containerapp create --name
                       --resource-group
                       [--allow-insecure {false, true}]
                       [--args]
                       [--bind]
                       [--command]
                       [--container-name]
                       [--cpu]
                       [--dal]
                       [--dapr-app-id]
                       [--dapr-app-port]
                       [--dapr-app-protocol {grpc, http}]
                       [--dapr-http-max-request-size]
                       [--dapr-http-read-buffer-size]
                       [--dapr-log-level {debug, error, info, warn}]
                       [--enable-dapr {false, true}]
                       [--env-vars]
                       [--environment]
                       [--environment-type {connected, managed}]
                       [--exposed-port]
                       [--image]
                       [--ingress {external, internal}]
                       [--max-replicas]
                       [--memory]
                       [--min-replicas]
                       [--no-wait]
                       [--registry-identity]
                       [--registry-password]
                       [--registry-server]
                       [--registry-username]
                       [--revision-suffix]
                       [--revisions-mode {multiple, single}]
                       [--scale-rule-auth]
                       [--scale-rule-http-concurrency]
                       [--scale-rule-metadata]
                       [--scale-rule-name]
                       [--scale-rule-type]
                       [--secret-volume-mount]
                       [--secrets]
                       [--system-assigned]
                       [--tags]
                       [--target-port]
                       [--termination-grace-period]
                       [--transport {auto, http, http2, tcp}]
                       [--user-assigned]
                       [--workload-profile-name]
                       [--yaml]
```

# Creating Container App using YAML file

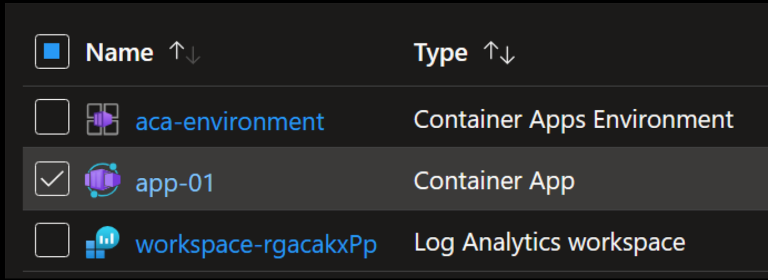
```
$ az containerapp create -n app
      -g rg --yaml app.yaml
```

```yaml
type: Microsoft.App/containerApps
tags:
  env: production
properties:
  managedEnvironmentId: <env_id>
  configuration:
    activeRevisionsMode: Multiple
    ingress:
      external: true
      allowInsecure: false
      targetPort: 80
  template:
    revisionSuffix: myrevision
    containers:
      - image: nginx
        name: nginx
```

```
$ kubectl apply -f deployment.yaml
```

Similar  to deploying Kubernetes YAML manifest.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp
spec:
  selector:
    matchLabels:
      app: webapp
  replicas: 5
  template:
    spec:
      containers:
      - name: webapp
        image: nginx
        ports:
        - containerPort: 80
```



| ■ | Name ↑↓ | Type ↑↓ |
|---|---|---|
| □ | aca-environment | Container Apps Environment |
| ☑ | app-01 | Container App |
| □ | workspace-rgacakxPp | Log Analytics workspace |

# Creating Container App using YAML file

```
$ az containerapp create -n app
     -g rg --yaml app.yaml
```

```yaml
type: Microsoft.App/containerApps
tags:
  env: production
properties:
  managedEnvironmentId: <env_id>
  configuration:
    activeRevisionsMode: Multiple
    ingress:
      external: true
      allowInsecure: false
      targetPort: 80
  template:
    revisionSuffix: myrevision
    containers:
      - image: nginx
        name: nginx
```

```
$ kubectl apply -f deployment.yaml
```

Similar  to deploying Kubernetes YAML manifest

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp
spec:
  selector:
    matchLabels:
      app: webapp
  replicas: 5
  template:
    spec:
      containers:
      - name: webapp
        image: nginx
        ports:
        - containerPort: 80
```

```
$ terraform apply -auto-approve
```

Similar to Terraform

```hcl
resource "azurerm_container_app" "aca" {
  name                         = "aca"
  container_app_environment_id = var.id
  resource_group_name          = rg.name
  revision_mode                = "Single"
  template {
    container {
      name   = nginx
      image  = nginx
    }
  }
  ingress {
    allow_insecure_connections = false
    external_enabled           = true
    target_port                = 80
    transport                  = "auto"
}
```

# Creating ACA App using YAML file

```yaml
location: canadacentral
name: mycontainerapp
resourceGroup: myresourcegroup
type: Microsoft.App/containerApps
tags:
  tagname: value
properties:
  managedEnvironmentId: env_id
  configuration:
    activeRevisionsMode: Multiple
    secrets:
      - name: mysecret
        value: thisismysecret
      - name: myregistrypassword
        value: I<3containerapps
    ingress:
      external: true
      allowInsecure: false
      targetPort: 80
      traffic:
        - latestRevision: true
          weight: 100
      transport: Auto
```

```yaml
  registries:
    - passwordSecretRef: pwd
      server: acr.azurecr.io
      username: myregistry
  dapr:
    appId: mycontainerapp
    appPort: 80
    appProtocol: http
    enabled: true
template:
  revisionSuffix: myrevision
  containers:
    - image: nginx
      name: nginx
      env:
        - name: HTTP_PORT
          value: 80
        - name: secret_name
          secretRef: mysecret
      command:
        - npm
        - start
      resources:
        cpu: 0.5
```

```yaml
probes:
  - type: liveness
    httpGet:
      path: "/health"
      port: 8080
      httpHeaders:
        - name: "Custom-Header"
          value: "liveness probe"
    initialDelaySeconds: 7
    periodSeconds: 3
  - type: readiness
    tcpSocket:
      port: 8081
    initialDelaySeconds: 10
    periodSeconds: 3
  - type: startup
    httpGet:
      path: "/startup"
      port: 8080
      httpHeaders:
        - name: "Custom-Header"
          value: "startup probe"
    initialDelaySeconds: 3
    periodSeconds: 3
volumeMounts:
  - mountPath: /myempty
    volumeName: myempty
  - mountPath: /myfiles
```