



# CKAD<sup>Q&As</sup>

Certified Kubernetes Application Developer (CKAD) Program

## Pass Linux Foundation CKAD Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.passapply.com/ckad.html>

100% Passing Guarantee  
100% Money Back Assurance

Following Questions and Answers are all new published by Linux Foundation Official Exam Center

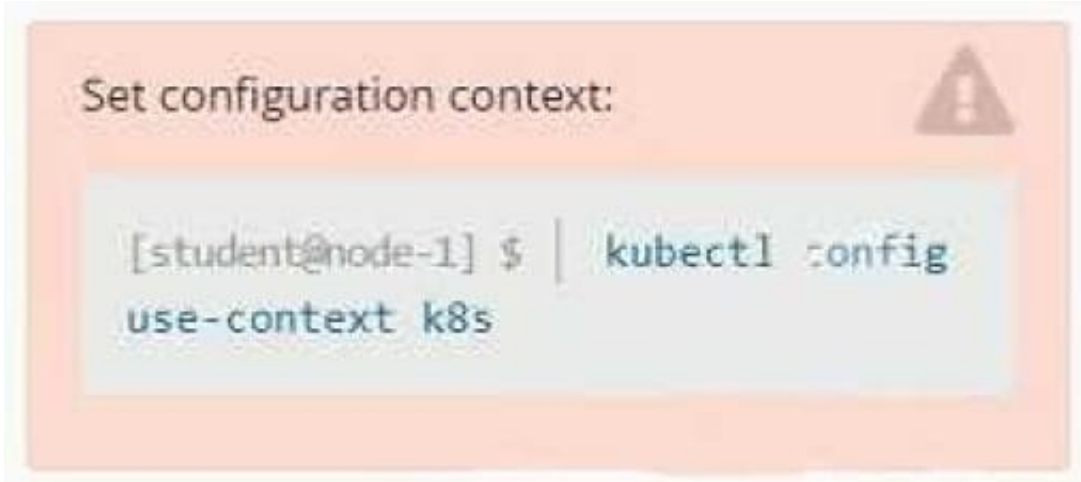
-  **Instant Download** After Purchase
-  **100% Money Back** Guarantee
-  **365 Days** Free Update
-  **800,000+** Satisfied Customers





## QUESTION 1

### CORRECT TEXT



#### Context

A pod is running on the cluster but it is not responding.

#### Task

The desired behavior is to have Kubernetes restart the pod when an endpoint returns an HTTP 500 on the /healthz endpoint. The service, probe-pod, should never send traffic to the pod while it is failing. Please complete the following:

1.

The application has an endpoint, /started, that will indicate if it can accept traffic by returning an HTTP 200. If the endpoint returns an HTTP 500, the application has not yet finished initialization.

2.

The application has another endpoint /healthz that will indicate if the application is still working as expected by returning an HTTP 200. If the endpoint returns an HTTP 500 the application is no longer responsive.

3.

Configure the probe-pod pod provided to use these endpoints

4.

The probes should use port 8080

A. Please check explanations

B. Place Holder

Correct Answer: A

apiVersion: v1



kind: Pod

metadata:

labels:

test: liveness

name: liveness-exec

spec:

containers:

-name: liveness

image: k8s.gcr.io/busybox

args:

-/bin/sh

- -c

-touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600 livenessProbe:

exec:

command:

-cat

-/tmp/healthy

initialDelaySeconds: 5

periodSeconds: 5

In the configuration file, you can see that the Pod has a single Container. The periodSeconds field specifies that the kubelet should perform a liveness probe every 5 seconds. The initialDelaySeconds field tells the kubelet that it should wait 5

seconds before performing the first probe. To perform a probe, the kubelet executes the command `cat /tmp/healthy` in the target container. If the command succeeds, it returns 0, and the kubelet considers the container to be alive and healthy.

If the command returns a non-zero value, the kubelet kills the container and restarts it.

When the container starts, it executes this command:

```
/bin/sh -c "touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600" For the first 30 seconds of the container's life, there is a /tmp/healthy file. So during the first 30 seconds, the command cat /tmp/healthy returns a success code. After 30
```

seconds, `cat /tmp/healthy` returns a failure code.

Create the Pod:



kubectl apply -f <https://k8s.io/examples/pods/probe/exec-liveness.yaml> Within 30 seconds, view the Pod events:

kubectl describe pod liveness-exec

The output indicates that no liveness probes have failed yet:

```

FirstSeen LastSeen Count From SubobjectPath Type Reason Message -----
----- 24s 24s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to worker0

23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox"

23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox"

23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined] 23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started
container

```

with docker id 86849c15382e

After 35 seconds, view the Pod events again:

kubectl describe pod liveness-exec

At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated. FirstSeen LastSeen Count From SubobjectPath Type Reason Message -----

```

----- 37s 37s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to
worker0

```

```

36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox"

```

```

36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox"

```

```

36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined] 36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started
container

```

with docker id 86849c15382e

```

2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open
\$/tmp/healthy\$: No such file or directory Wait another 30 seconds, and verify that the container has been restarted:

```

kubectl get pod liveness-exec

The output shows that RESTARTS has been incremented:

```

NAME READY STATUS RESTARTS AGE

```

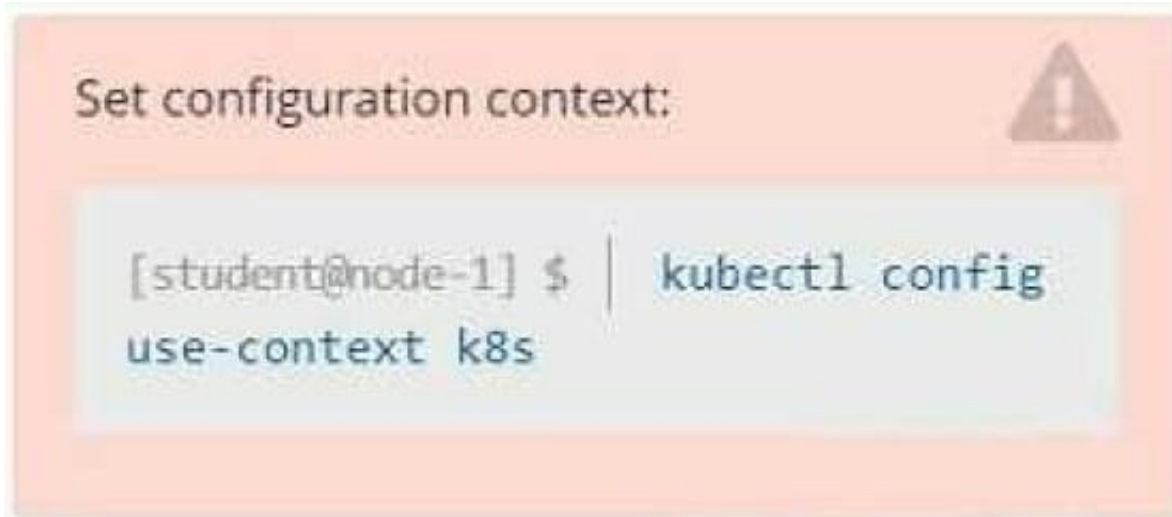
```

liveness-exec 1/1 Running 1 1m

```

## QUESTION 2

CORRECT TEXT



### Context

Developers occasionally need to submit pods that run periodically.

### Task

Follow the steps below to create a pod that will start at a predetermined time and]which runs to completion only once each time it is started:

Create a YAML formatted Kubernetes manifest `/opt/KDPD00301/periodic.yaml` that runs the following shell command: `date` in a single busybox container.

The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be `hello`

Create the resource in the above manifest and verify that the job executes successfully at least once

A. Please check explanations

B. Place Holder

Correct Answer: A



```
Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```

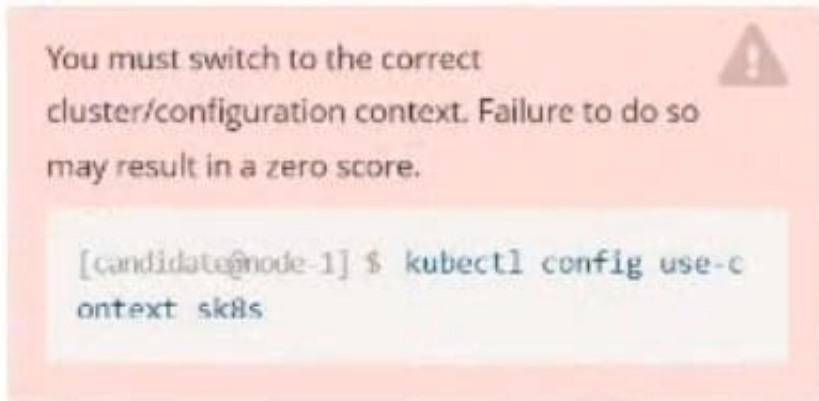
```
Readme Web Terminal THE LINUX FOUNDATION
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-o", "date"]
              restartPolicy: Never
          schedule: */1 * * * *
          startingDeadlineSeconds: 22
          concurrencyPolicy: Allow
```

```
Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME          SCHEDULE          SUSPEND   ACTIVE   LAST SCHEDULE   AGE
hello        */1 * * * *      False    0        <none>           6s
student@node-1:~$
```



### QUESTION 3

CORRECT TEXT



Task:

Modify the existing Deployment named broker-deployment running in namespace quetzal so that its containers.

The broker-deployment is manifest file can be found at:

```
~/daring_mocassin/broker-deployment.yaml
```

- A. Please check explanations
- B. Place Holder

Correct Answer: A





```
candidate@node-1:~$ kubectl config use-context k8s  
Switched to context "k8s".  
candidate@node-1:~$ vim
```

File Edit View Terminal Tabs Help

```
containers:  
- name: broker  
  image: redis:alpine  
  ports:  
  - containerPort: 6379  
  securityContext:  
    runAsUser: 30000  
    privileged: false
```

:wq

```
candidate@node-1:~$ kubectl config use-context k8s  
Switched to context "k8s".  
candidate@node-1:~$ vim ~/daring-moccasin/broker-deployment.yaml  
candidate@node-1:~$ kubectl apply -f ~/daring-moccasin/broker-deployment.yaml  
deployment.apps/broker-deployment configured  
candidate@node-1:~$ kubectl get pods -n quetzal  
NAME                                READY   STATUS    RESTARTS   AGE  
broker-deployment-65446d6d94-868p6  1/1     Running   0           30s  
broker-deployment-65446d6d94-8dn7l  1/1     Running   0           32s  
broker-deployment-65446d6d94-p4h4l  1/1     Running   0           31s  
candidate@node-1:~$ kubectl get deploy -n quetzal  
NAME                READY   UP-TO-DATE   AVAILABLE   AGE  
broker-deployment  3/3     3             3           7h3m  
candidate@node-1:~$
```

#### QUESTION 4

CORRECT TEXT

No configuration context change is required for this task.

Task:

A Dockerfile has been prepared at `~/human-stork/build/Dockerfile`





Multiple image builders and tools have been pre-installed in the base system, including: docker, skopeo, buildah, img, and podman.

Please do not push the built image to a registry, run a container, or otherwise consume it.

A. Please check explanations

B. Place Holder

Correct Answer: A

```
candidate@node-1:~$ cd humane-stork/build/
candidate@node-1:~/humane-stork/build$ ls -l
total 16
-rw-r--r-- 1 candidate candidate 201 Sep 24 04:21 Dockerfile
-rw-r--r-- 1 candidate candidate 644 Sep 24 04:21 text1.html
-rw-r--r-- 1 candidate candidate 813 Sep 24 04:21 text2.html
-rw-r--r-- 1 candidate candidate 383 Sep 24 04:21 text3.html
candidate@node-1:~/humane-stork/build$ sudo docker build -t macaque:3.0 .
Sending build context to Docker daemon 6.144kB
Step 1/5 : FROM docker.io/lfccncf/nginx:mainline
--> ea335eea17ab
Step 2/5 : ADD text1.html /usr/share/nginx/html/
--> 8967ee9ee5d0
Step 3/5 : ADD text2.html /usr/share/nginx/html/
--> cb0554422f26
Step 4/5 : ADD text3.html /usr/share/nginx/html/
--> 62e879ab821e
Step 5/5 : COPY text2.html /usr/share/nginx/html/index.html
--> 331c8a94372c
Successfully built 331c8a94372c
Successfully tagged macaque:3.0
candidate@node-1:~/humane-stork/build$ sudo docker save macaque:3.0 > ~/humane-stork/macaque-3.0.tar
candidate@node-1:~/humane-stork/build$ cd ..
candidate@node-1:~/humane-stork$ ls -l
total 142532
drwxr-xr-x 2 candidate candidate 4096 Sep 24 04:21 build
-rw-rw-r-- 1 candidate candidate 145948672 Sep 24 11:39 macaque-3.0.tar
candidate@node-1:~/humane-stork$
```

## QUESTION 5

CORRECT TEXT



### Context

It is always useful to look at the resources your applications are consuming in a cluster.

### Task

From the pods running in namespace `cpu-stress`, write the name only of the pod that is consuming the most CPU to file `/opt/KDOBG030/pod.txt`, which has already been created.

- A. Please check explanations
- B. Place Holder

Correct Answer: A

[Latest CKAD Dumps](#)

[CKAD Practice Test](#)

[CKAD Study Guide](#)