Red Hat Openshift Developer EX288 Practice Exam (Sample)

Study Guide

Everything you need from our exam experts!

Table of Contents

Copyright **1**

Table of Contents **2**

Introduction **3**

How to Use This Guide **4**

Questions **5**

Answers **9**

Explanations **11**

Next Steps **18**

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- **Practice answering questions under realistic conditions,**
- **Improve accuracy and speed,**
- **Review explanations to strengthen weak areas, and**
- **Approach the exam with greater confidence.**

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here’s how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don’t just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There’s no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

Questions

1. **In Tekton, which component executes the defined Tasks to perform the build and deploy steps?**

   A. Pipeline

   B. PipelineResource

   C. PipelineRun

   D. Task

2. **Which command retrieves the tags for all inbuilt PHP image streams in OpenShift?**

   A. oc get istag -n openshift | grep php

   B. oc get istag -n openshift | grep php7

   C. oc get imagestreamtags -n openshift

   D. oc get istag -n openshift

3. **How do you update an existing Deployment to use a new image tag?**

   A. Delete the deployment and recreate with the new image tag.

   B. Edit the Deployment YAML manually and apply changes.

   C. Use oc set image deployment/<name> <container>=<image:tag>, or edit YAML and apply, or trigger ImageChange.

   D. Rebuild the application and redeploy the entire cluster.

4. **How is a ServiceAccount's API token made available to applications inside a pod, and how can an application use it?**

   A. Tokens are embedded in the container images.

   B. The token is mounted as a file in the pod’s filesystem under the service account; applications can use it to authenticate to the API from within the cluster.

   C. Tokens are retrieved via an external service on startup.

   D. API tokens are not used for in-cluster authentication.

5. How do you define an overlay file for a deployment to override the replica count to 5?

A. replicas: 5 at the Deployment level in a manifest

B. apiVersion: apps/v1

kind: Deployment

metadata:

name: myapp

spec:

replicas: 5

C. spec: replicas: 5

D. kind: Service

metadata:

name: myapp

spec:

replicas: 5

6. How does a Route work in OpenShift, and what TLS termination options are available?

A. Route exposes an internal service only within the cluster.

B. Route exposes a service to external clients; TLS termination options include edge, reencrypt, and passthrough.

C. Route is used to configure persistent storage.

D. Route creates a new deployment.

7. Which statement best describes the purpose of values.yaml in a Helm chart?

A. It defines the chart's metadata and dependencies.

B. It lists all resources to deploy in the cluster.

C. It holds configuration values for the chart that can be overridden.

D. It stores release history.

8. View the rollout history for a deployment config php.

A. oc rollout history dc/php

B. oc rollout status dc/php

C. oc rollout history deployment/php

D. oc logs dc/php

9. Compare Deployment and StatefulSet in OpenShift: when would you choose each?

A. Use Deployment for stateless apps; StatefulSet for stateful apps requiring stable identities and persistent storage.

B. Use StatefulSet for stateless; Deployment for stateful.

C. Use Deployment for long-running jobs; StatefulSet for ephemeral.

D. Use Deployment for secrets; StatefulSet for config maps.

10. Which command starts a build for an application?

A. oc start-build

B. oc start-build myapp

C. oc start-build myapp --watch

D. oc build-start

Answers

1. C
2. A
3. C
4. B
5. B
6. B
7. C
8. A
9. A
10. A

Explanations

1. In Tekton, which component executes the defined Tasks to perform the build and deploy steps?

A. Pipeline

B. PipelineResource

C. PipelineRun

D. Task

In Tekton, a PipelineRun is the runtime instance that drives the work defined in a Pipeline. A Pipeline groups together Tasks, each Task describing the steps to perform, while a PipelineRun creates and coordinates the execution of those Tasks. When a PipelineRun starts, Tekton creates TaskRun resources for each Task in the Pipeline and runs them in the required order or in parallel as defined. This orchestration is what actually carries out the build and deploy steps. The Task itself defines what to do, but it's the PipelineRun that executes those Tasks by launching and coordinating their TaskRuns.

2. Which command retrieves the tags for all inbuilt PHP image streams in OpenShift?

A. oc get istag -n openshift | grep php

B. oc get istag -n openshift | grep php7

C. oc get imagestreamtags -n openshift

D. oc get istag -n openshift

Retrieving the PHP image stream tags starts with listing the image stream tags in the OpenShift openshift namespace and then filtering for PHP. The istag resource represents individual tags of image streams, so using oc get istag -n openshift shows all tags, and piping through grep php narrows the output to only the PHP-related streams. This directly provides the tags for all built-in PHP image streams. Listing all image stream tags without filtering would include many non-PHP streams, and filtering only for php7 could miss other PHP tags.

3. How do you update an existing Deployment to use a new image tag?

A. Delete the deployment and recreate with the new image tag.

B. Edit the Deployment YAML manually and apply changes.

C. Use oc set image deployment/<name> <container>=<image:tag>, or edit YAML and apply, or trigger ImageChange.

D. Rebuild the application and redeploy the entire cluster.

Updating a Deployment to use a new image tag is done by performing a rolling update on the Deployment's Pod template. This means changing the image reference for the container in the Deployment so that Kubernetes/OpenShift gradually replaces old pods with new ones running the updated image, without taking the app offline. You can do this in a few straightforward ways. One common method is using the OpenShift CLI: oc set image deployment/<name> <container>=<image:tag>. This updates the image for that container and triggers a new rollout that replaces pods with the new version. Another valid approach is to edit the Deployment YAML (specifically spec.template.spec.containers[].image) to the new image:tag and apply the changes; this also starts a rolling update. You can also configure an ImageChange trigger so that an update to the image stream automatically triggers a deployment, enabling automatic rollouts in CI/CD workflows. Deleting the deployment and recreating it is not desirable because it disrupts the running application and bypasses the controlled rolling update process. Rebuilding the entire cluster is unnecessary for updating a single deployment's image.

4. How is a ServiceAccount's API token made available to applications inside a pod, and how can an application use it?

A. Tokens are embedded in the container images.

B. The token is mounted as a file in the pod's filesystem under the service account; applications can use it to authenticate to the API from within the cluster.

C. Tokens are retrieved via an external service on startup.

D. API tokens are not used for in-cluster authentication.

Inside a pod, applications authenticate to the cluster's API using a token that is provided by the Pod's ServiceAccount. This token isn't baked into the container image; instead, the cluster automatically mounts a secret for the ServiceAccount into the pod's filesystem. The application can then read that token file and use it to authenticate requests to the API server from within the cluster. In practice, the token appears as a file inside the pod, commonly at a path like /var/run/secrets/kubernetes.io/serviceaccount/token, with the CA certificate and namespace also exposed in the same mounted secret. The application can pass the token as a Bearer token in HTTP requests when talking to the API, or rely on client libraries that are configured to use in-cluster credentials and automatically read this token from the standard path. This approach avoids embedding credentials in container images and enables secure, automated authentication for in-cluster components.

5. How do you define an overlay file for a deployment to override the replica count to 5?

A. replicas: 5 at the Deployment level in a manifest

B. apiVersion: apps/v1

kind: Deployment

metadata:

name: myapp

spec:

replicas: 5

C. spec: replicas: 5

D. kind: Service

metadata:

name: myapp

spec:

replicas: 5

Overlays and patches modify specific fields of an existing resource, so you need a patch that clearly identifies which resource to change and what to change. To override the number of replicas for a Deployment, you patch the Deployment by including its identifying details and the new replica value under spec.replicas. The YAML shown contains all the necessary parts: apiVersion and kind specify the resource type, metadata.name identifies the exact Deployment to modify, and spec.replicas sets the desired count to 5. This makes it a valid overlay patch for that Deployment, reliably applying the change to the correct resource. The other options aren't suitable because they're incomplete, not patch-like, or refer to a different resource type (which doesn't use replicas).

6. How does a Route work in OpenShift, and what TLS termination options are available?

A. Route exposes an internal service only within the cluster.

B. Route exposes a service to external clients; TLS termination options include edge, reencrypt, and passthrough.

C. Route is used to configure persistent storage.

D. Route creates a new deployment.

A Route in OpenShift is used to expose a service to external clients, providing a hostname and rules that tell the cluster's router how to send traffic from outside to the internal service. TLS termination options indicate where the secure connection is terminated and how encryption is handled between the router and the backend: - Edge termination: the TLS handshake ends at the router. The router decrypts the client traffic and forwards it to the service, typically over HTTP. The router supplies the certificate that clients connect to. - Reencrypt termination: TLS ends at the router, but the router then re-encrypts traffic and sends it to the backend, using another certificate for the backend connection. This keeps end-to-end encryption between client and service, with the router handling TLS on both sides. - Passthrough termination: the TLS connection is not terminated at the router at all; the client's TLS traffic goes straight through to the backend, and the service handles TLS itself. So, a Route exposes a service to external clients, and you can choose how TLS is handled: at the edge (router), reencrypted to the backend, or passed through to the backend.

7. Which statement best describes the purpose of values.yaml in a Helm chart?

A. It defines the chart's metadata and dependencies.

B. It lists all resources to deploy in the cluster.

C. It holds configuration values for the chart that can be overridden.

D. It stores release history.

Configuration values that drive the chart templates are provided by values.yaml. In a Helm chart, templates reference variables under .Values, so the defaults you put in values.yaml are used to render the final Kubernetes manifests. This setup makes the chart reusable across environments because you can customize behavior without editing the templates themselves. For example, you might define replicaCount and image.tag in values.yaml; when you install or upgrade, you can override them with --values myvalues.yaml or --set replicaCount=5 to produce a different deployment. The other aspects of a chart live elsewhere: chart metadata and dependencies are in Chart.yaml, the actual resources are created by the templates, and release history is managed by Helm, not stored in values.yaml.

8. View the rollout history for a deployment config php.

A. oc rollout history dc/php

B. oc rollout status dc/php

C. oc rollout history deployment/php

D. oc logs dc/php

To view how a DeploymentConfig has rolled out over time, use the rollout history command on the DeploymentConfig resource. For a DeploymentConfig named php, the correct syntax is oc rollout history dc/php. This shows each rollout revision with details like the image and change cause, letting you see what changed and decide if you need to revert. The other commands don't fit: oc rollout status dc/php shows the current rollout progress rather than the full history; oc rollout history deployment/php would target a Kubernetes Deployment resource, not an OpenShift DeploymentConfig; and oc logs dc/php would display logs, not rollout history.

9. Compare Deployment and StatefulSet in OpenShift: when would you choose each?

A. Use Deployment for stateless apps; StatefulSet for stateful apps requiring stable identities and persistent storage.

B. Use StatefulSet for stateless; Deployment for stateful.

C. Use Deployment for long-running jobs; StatefulSet for ephemeral.

D. Use Deployment for secrets; StatefulSet for config maps.

The choice hinges on stateless versus stateful workloads. Deployments are ideal for stateless apps where each pod can be freely replaced, scaled, or updated without needing to preserve identity or local state. They don't guarantee a stable network identity or persistent storage across restarts, which keeps them simple and highly scalable for front-end services, APIs, and other non-stateful components. StatefulSets, on the other hand, are designed for stateful workloads that must keep stable identities and durable storage. Each pod gets a predictable, unique name and a stable network identity, and you can attach persistent volumes that remain associated with that pod across rescheduling. This makes StatefulSets suitable for databases, caches, and other systems where preserving state and identity is essential, and where ordered deployment, scaling, and upgrades are important. So, use a Deployment for stateless applications; use a StatefulSet for stateful applications requiring stable identities and persistent storage. The other options don't fit because using StatefulSet for stateless workloads adds unnecessary complexity, Deployments aren't the pattern for long-running jobs (Jobs/CronJobs are), and secrets/config maps aren't a differentiator between these two controllers.

10. Which command starts a build for an application?

A. oc start-build

B. oc start-build myapp

C. oc start-build myapp --watch

D. oc build-start

Starting a build in OpenShift is done with the start-build command. This is the direct action that tells the cluster to kick off a new build for a BuildConfig. The base form oc start-build represents that trigger. If you already know the specific BuildConfig (for example, the app's BuildConfig name), you can target it explicitly by adding that name (oc start-build myapp). The --watch option simply streams the build logs as it runs, and oc build-start isn't a valid OpenShift CLI command. So the most general and correct command to initiate a build is oc start-build.

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://redhatopenshiftdevex288.examzify.com

We wish you the very best on your exam journey. You've got this!