Red Hat Certified Cloud-native Developer exam (EX378)

ID EX378 **Preis** CHF 557.– (exkl. MwSt.) **Dauer** 0.4 Tage

Zielgruppe

Die Zertifizierung als Red Hat Certified Cloud-native Developer kommt für folgende Zielgruppen in Frage:

- Java-Entwickler, die Microservices mit Quarkus und Kubernetes implementieren
- Red Hat Certified Professionals, die die Zertifizierung als Red Hat Certified Architect (RHCA) erhalten möchten

Voraussetzungen

- Teilnahme am Kurs Red Hat Cloud-native Microservices Development with Quarkus (DO378) oder vergleichbare praktische Erfahrungen
- Kenntnisse in der Verwendung von Visual Code/Codium in einer Red Hat Enterprise Linux Umgebung
- Umfangreiche Erfahrung mit JSE, einschliesslich Kenntnissen in den wichtigsten Java-Konzepten und APIs Beispielsweise werden für die Prüfung die Exceptions, Annotations und Collections API benötigt.
- Grundkenntnisse in OpenShift/Kubernetes sind von Vorteil.

Als Vorbereitung

Red Hat empfiehlt zur Vorbereitung den Kurs Red Hat Cloud-native Microservices Development with Quarkus (DO378). Die Teilnahme an diesem Kurs ist nicht vorgeschrieben. Es kann auch nur die Prüfung abgelegt werden.

Auch wenn die Teilnahme an Red Hat Kursen einen wichtigen Teil der Prüfungsvorbereitung darstellt, ist sie keine Garantie für das Bestehen der Prüfung. Vorherige Erfahrung, Praxis und Eignung sind darüber hinaus wichtige Erfolgsfaktoren.

Zur Systemadministration für Produkte von Red Hat sind zahlreiche Bücher und andere Ressourcen erhältlich. Eine offizielle Empfehlung zur Nutzung solcher Materialien für die Vorbereitung auf die Prüfungen gibt Red Hat jedoch nicht. Dennoch kann sich weiterführende Literatur stets als hilfreich erweisen.

Kursinhalt

Lerninhalte für die Prüfung

Um Sie bei der Vorbereitung zu unterstützen, haben wir nachfolgend die Prüfungsziele und Aufgabenbereiche aufgelistet, die in der Prüfung abgefragt werden. Red Hat behält sich das Recht vor, Prüfungsziele hinzuzufügen, zu ändern oder zu entfernen. Solche Änderungen werden im Voraus bekannt gegeben.

Die Teilnehmer sollten mit folgenden Aufgaben vertraut sein:

Angabe und Abruf von Konfigurationseigenschaften über verschiedene umgebungsbewusste Quellen und Bereitstellung per Dependency Injection oder Lookup

- Daten in konfigurierten Werten externalisieren
- Konfigurierte Werte mit dem Qualifier @Inject und @ConfigProperty in Beans einfügen
- Auf Konfigurationen zugreifen oder diese erstellen
- Standardmässiges und benutzerdefiniertes ConfigSource und ConfigSource-Ordering verstehen

Entwicklung fehlertoleranter Quarkus-basierter Microservices mit MicroProfile Fault Tolerance-Strategien

- Beziehung zu MicroProfile Config verstehen
- Ausführungstypen async und sync verstehen
- @Timeout nutzen
- Retry-Richtlinien und ihre Anwendung mit @Retry verstehen
- Fallback verstehen und definieren
- CircuitBreaker verstehen und anwenden
- Bulkhead verstehen und anwenden
- Fault Tolerance-Konfigurationen verstehen und einrichten

Prüfung des Status von Quarkus-Anwendungen über einen anderen Rechner mit MicroProfile Health Check

- HealthCheck-Oberfläche verstehen und implementieren
- @Liveness- und @Readiness-Annotation verstehen und anwenden
- HealthCheck-Oberfläche verstehen und implementieren
- Benutzerfreundliche HealthCheckResponse erstellen

Export von Überwachungsdaten an Management-Agents aus

einer Quarkus-Anwendung mit MicroProfile Metrics

- Drei Gruppen von untergeordneten Ressourcen (Scopes) verstehen und anwenden: Base, Vendor und Application
- Tags (Labels) verstehen
- Metadaten verstehen und nutzen
- Metric Registry und @Metric verstehen
- Kennzahlen per REST-API freigeben
- Erforderliche Kennzahlen kennen
- Programmiermodell für Quarkus-Anwendungskennzahlen verstehen

MicroProfile Interoperable JWT RBAC in Quarkus-Anwendungen: OIDC-basierte (OpenID Connect) JSON Web Tokens (JWTs) für RBAC (Role-Based Access Control) von Microservice-Endpunkten

- Tokenbasierte Authentifizierung verstehen
- JWT Bearer Tokens zum Schutz von Services verwenden
- JAX-RS-Anwendung so markieren, dass die Zugriffskontrolle MP-JWT verwendet werden muss
- MP-JWT-Tokens und Java EE Container APIs einander zuordnen

Implementierung einer Quarkus-Anwendung und Freigabe von REST-Serviceendpunkten mit JAX-RS

- REST-Konzepte verstehen, insbesondere die Anwendung und Nutzung von HTTP PUT, DELETE, GET und POST-Methoden
- Standardmässige HTTP-Return-Codes kennen und verwenden
- RESTful Root-Ressourcenklassen implementieren
- REST-Services mit JAX-RS freigeben
- @Path, @Produce und @Consume verstehen und verwenden
- CDI zur Integration von Komponenten verwenden
- Bean Validation zur Sicherstellung von Datenformat und Konsistenz verwenden

Vereinfachte JPA-Zuordnung mit Panache

- Den Unterschied zwischen dem Active Record Pattern und dem Repository Pattern verstehen
- JPA verwenden, um persistente Objekte und deren Beziehungen zu erstellen, zu lesen, zu aktualisieren und zu löschen
- Eine bidirektionale One-to-many-Beziehung zwischen zwei Entitäten zuordnen, einschliesslich der beiden Seiten der Beziehung
- Die gängigsten Panache-Vorgänge durchführen und benutzerdefinierte Entitätsmethoden hinzufügen können

MicroProfile OpenAPI-Spezifikation zur Dokumentation von RESTful APIs

- OpenAPI-Dokumente und die Swagger-Benutzeroberfläche verstehen, um Remote-Services-APIs zu entdecken
- Remote-Service-Endpunkte für semantische Versionsverwaltung (Semver) verlinken können
- Standardmässige und benutzerdefinierte OpenAPI-Dokumente auf JAX-RS-Endpunkten erstellen können

Interaktion mit REST-APIs in Quarkus über den MicroProfile REST Client

- Den typsicheren Ansatz zum Aufrufen von RESTful Services über HTTP mit JAX-RS APIs verstehen
- REST-Konzepte verstehen, insbesondere die Anwendung und Nutzung von HTTP PUT, DELETE, GET und POST-Methoden
- Einen REST-Client für die Verbindung mit einem Remote-Service erstellen und verwenden können
- REST-Client-URIs für den Aufruf von bestimmten Remote-Microservices parametrisieren und konfigurieren
- Spezielle zusätzliche Client-Header verstehen und verwenden

Wie bei allen leistungsbasierten Red Hat Prüfungen müssen die Konfigurationen nach einem Neustart ohne Eingreifen bestehen bleiben.

Prüfungsformat

Das Red Hat Certified Cloud-Native Developer Exam ist eine praxisorientierte Prüfung, in der Sie reale Aufgaben lösen. Während der Prüfung besteht kein Zugang zum Internet, und es dürfen keine gedruckten oder elektronischen Dokumente zur Prüfung mitgebracht werden. Dieses Verbot schliesst auch Notizen, Bücher oder sonstige Materialien ein. Bei den meisten Prüfungen steht die im Produktumfang enthaltene Dokumentation zur Verfügung.

Bekanntgabe von Prüfungsergebnissen

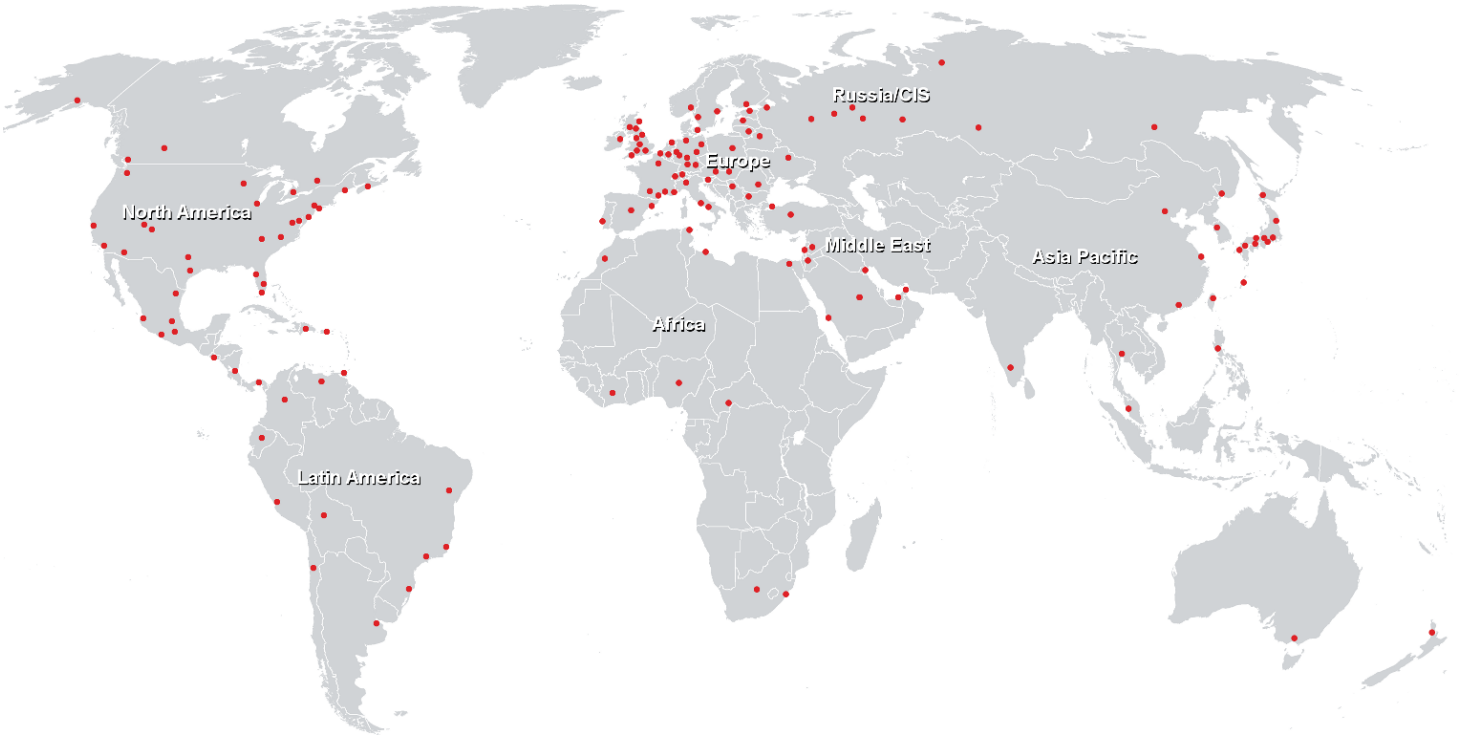
Die offiziellen Prüfungsergebnisse werden ausschliesslich auf der Website Red Hat Certification Central veröffentlicht. Red Hat erlaubt Prüfern oder Trainingspartnern nicht, den Teilnehmern die Ergebnisse direkt mitzuteilen. In der Regel wird das Ergebnis innerhalb von drei US-Werktagen mitgeteilt.

Die Prüfungsergebnisse werden in Form einer Gesamtpunktzahl kommuniziert. Red Hat erteilt keine Informationen über einzelne Prüfungselemente und gibt auch auf Anfrage keine weiteren

Auskünfte.

Red Hat Certified Cloud-native Developer exam (EX378)

Weltweite Trainingscenter



Fast Lane Institute for Knowledge Transfer (Switzerland) AG

Husacherstrasse 3
CH-8304 Wallisellen
Tel. +41 44 832 50 80

info@flane.ch, https://www.flane.ch