

Anwendungsstartprobleme in WebSphere Liberty Profile

Es scheint, dass nach dem Herunterstufen Ihrer Spring Boot-Anwendung auf Version 2.7.18 und dem Bereitstellen auf WebSphere Liberty Profile (WLP) ein Problem auftritt, bei dem die Anwendung nicht innerhalb der Standardzeitüberschreitung startet, wie in der `console.log`-Datei angegeben: *“die Anwendung myapp hat nicht innerhalb von 30.xx Sekunden gestartet”*. Dies deutet darauf hin, dass die Anwendung entweder zu lange zum Initialisieren benötigt oder überhaupt nicht startet. Im Folgenden werde ich Sie durch einen schrittweisen Prozess führen, um dieses Problem zu diagnostizieren und zu beheben.

Schritt 1: Serverprotokolle auf Hinweise untersuchen

Zuerst sollten Sie die Serverprotokolle genauer untersuchen, um die Ursache des Startfehlers zu identifizieren. Die `console.log`-Datei enthält die Zeitüberschreitungsmeldung, enthält aber möglicherweise nicht die ganze Geschichte. Überprüfen Sie die folgenden Protokolldateien im Protokollverzeichnis Ihres WLP-Servers (`/opt/ibm/wlp/usr/servers/myServer/logs/`):

- `messages.log`: Diese Datei enthält oft FEHLER- oder WARNUNGEN-Meldungen, die Probleme wie fehlende Abhängigkeiten, Konfigurationsfehler oder Ausnahmen während des Startens aufzeigen können.
- `trace.log`: Wenn detailliertes Tracing aktiviert ist, kann diese Datei mehr Kontext darüber bieten, was während der Bereitstellung passiert.

Achten Sie auf:

- Stacktraces oder Ausnahmen (z. B. `ClassNotFoundException`, `NoSuchBeanDefinitionException`).
- Meldungen über fehlende Ressourcen oder inkompatible Bibliotheken.
- Hinweise darauf, dass der Anwendungskontext nicht initialisiert wurde.

Wenn Sie nicht genug Details sehen, können Sie das Protokollierungsniveau in WLP erhöhen, indem Sie die `server.xml`-Datei ändern. Fügen Sie oder aktualisieren Sie das `<logging>`-Element wie folgt:

```
<logging traceSpecification="*=info:com.ibm.ws.webcontainer*=all" />
```

Starten Sie den Server nach dieser Änderung neu, stellen Sie Ihre Anwendung erneut bereit und überprüfen Sie die Protokolle erneut auf weitere Informationen.

Schritt 2: Anwendungsstart mit Protokollierung überprüfen

Da es sich um eine Spring Boot-Anwendung handelt, könnte das Problem mit der Initialisierung des Anwendungskontextes zusammenhängen. Um zu bestimmen, wie weit der Startprozess fortschreitet, fügen

Sie eine einfache Protokollierungsanweisung zu Ihrer Hauptanwendungsklasse mit einer `@PostConstruct`-Methode hinzu. Hier ist ein Beispiel:

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;
import javax.annotation.PostConstruct;

@SpringBootApplication
public class DemoApplication extends SpringBootServletInitializer {

    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
        return application.sources(DemoApplication.class);
    }

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

    @PostConstruct
    public void init() {
        System.out.println("Anwendungskontext initialisiert");
    }
}
```

- Bauen Sie Ihre Anwendung neu (`mvn clean package`).
- Stellen Sie die WAR-Datei im `dropins`-Verzeichnis von WLP bereit.
- Überprüfen Sie `console.log` auf die Meldung "Anwendungskontext initialisiert".

Wenn diese Meldung erscheint, wird der Anwendungskontext erfolgreich geladen, und das Problem könnte mit Webkomponenten oder der Servlet-Initialisierung zusammenhängen. Wenn sie nicht erscheint, tritt das Problem früher während der Kontextinitialisierung auf.

Schritt 3: Debug-Protokollierung in Spring Boot aktivieren

Um mehr Einblicke in den Startprozess von Spring Boot zu erhalten, aktivieren Sie die Debug-Protokollierung, indem Sie eine Konfigurationsdatei hinzufügen. Erstellen oder bearbeiten Sie `src/main/resources/application.properties` wie folgt:

```
debug=true
```

- Bauen und stellen Sie die Anwendung erneut bereit.
- Überprüfen Sie `console.log` (oder andere Protokolle) auf detaillierte Debug-Ausgaben von Spring Boot.

Dies protokolliert Informationen zur Bean-Erstellung, Autokonfiguration und etwaigen Fehlern, die während des Startens auftreten. Suchen Sie nach Hinweisen darauf, was möglicherweise hängen bleibt oder fehlschlägt.

Schritt 4: WAR-Datei und Abhängigkeitskonfiguration überprüfen

Da Sie auf WLP bereitstellen, das seinen eigenen Servlet-Container bereitstellt, stellen Sie sicher, dass Ihre WAR-Datei für einen externen Server korrekt konfiguriert ist:

- **WAR-Packaging:** In Ihrem `pom.xml` stellen Sie sicher, dass das Packaging auf `war` gesetzt ist:

```
<packaging>war</packaging>
```

- **Tomcat als bereitgestellt:** Stellen Sie sicher, dass der eingebettete Tomcat von der WAR-Datei ausgeschlossen wird, da WLP den Servlet-Container bereitstellt. Überprüfen Sie Ihr `pom.xml` auf:

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-web</artifactId>  
</dependency>  
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-tomcat</artifactId>  
  <scope>provided</scope>  
</dependency>
```

- **Servlet-API-Kompatibilität:** Spring Boot 2.7.18 verwendet `javax.servlet:javax.servlet-api:4.0.1`, das mit dem `javaee-8.0`-Feature von WLP (Servlet 4.0) kompatibel ist. Um sicherzustellen, dass keine widersprüchlichen Abhängigkeiten vorhanden sind, führen Sie aus:

```
mvn dependency:tree
```

Achten Sie auf unerwartete Servlet-API-Versionen (z. B. `jakarta.servlet-api`, die in Spring Boot 3.x verwendet wird und mit `javaee-8.0` nicht kompatibel ist).

Wenn Sie vermuten, dass Abhängigkeitsprobleme vorliegen, entpacken Sie die WAR-Datei und überprüfen Sie `WEB-INF/lib`, um sicherzustellen, dass keine unerwarteten Servlet-bezogenen JARs enthalten sind.

Schritt 5: Lokal testen, um das Problem zu isolieren

Um zu bestimmen, ob das Problem spezifisch für WLP oder die Anwendung selbst ist, testen Sie die Anwendung lokal mit dem eingebetteten Tomcat:

```
mvn spring-boot:run
```

Wenn sie erfolgreich startet und Sie auf Ihre Endpunkte zugreifen können (z. B. einen einfachen "Hello World!" REST-Controller), liegt das Problem wahrscheinlich in der WLP-Bereitstellung und nicht im Anwendungscode.

Schritt 6: WLP-Startzeitüberschreitung anpassen (vorläufige Lösung)

Wenn die Protokolle darauf hinweisen, dass die Anwendung startet, aber länger als 30 Sekunden benötigt, können Sie die Startzeitüberschreitung in der `server.xml` von WLP erhöhen:

```
<applicationMonitor startTimeout="60s" />
```

- Stellen Sie die Anwendung erneut bereit und überwachen Sie die Protokolle.
- Wenn sie nach der erweiterten Zeitüberschreitung startet, bestätigt dies einen langsamen Startprozess, und Sie sollten die Anwendung optimieren (z. B. die Komponentenscans oder Initialisierungstasks reduzieren).

Dies ist jedoch eine vorläufige Lösung –idealerweise sollte eine einfache Anwendung innerhalb von 30 Sekunden starten, daher sollten Sie die Ursache weiter untersuchen.

Schritt 7: Vereinfachen und mit einem neuen Projekt vergleichen

Wenn das Problem weiterhin besteht, erstellen Sie ein minimales Spring Boot 2.7.18-Projekt, um die Bereitstellung auf WLP zu testen: 1. Verwenden Sie Spring Initializr mit: - Spring Boot 2.7.18 - Java (entsprechend Ihrer WLP-Version, z. B. 8 oder 11) - Abhängigkeit: Spring Web 2. Fügen Sie einen grundlegenden REST-Controller hinzu:

```
package com.example.demo;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloController {
    @GetMapping("/")
    public String hello() {
        return "Hello World!";
    }
}
```

3. Konfigurieren Sie es für die WAR-Bereitstellung (erweitern Sie `SpringBootServletInitializer` wie oben gezeigt).
4. Bauen Sie die WAR-Datei (`mvn clean package`) und stellen Sie sie im `dropins`-Verzeichnis von WLP bereit.

Wenn dieses neue Projekt erfolgreich startet, vergleichen Sie dessen `pom.xml`, Hauptklasse und Konfiguration mit Ihrem ursprünglichen Projekt, um Unterschiede zu identifizieren, die zum Fehler führen.

Schritt 8: Zusätzliche WLP-Fehlerbehebung

Wenn keines der oben genannten Maßnahmen das Problem löst, überlegen Sie sich diese WLP-spezifischen Überprüfungen: - **Arbeitsverzeichnis bereinigen**: Entfernen Sie zwischengespeicherte Daten, die die Bereitstellung stören könnten: `bash rm -rf /opt/ibm/wlp/usr/servers/myServer/workarea` Starten Sie dann den Server neu und stellen Sie die Anwendung erneut bereit. - **WLP-Version überprüfen**: Führen Sie `/opt/ibm/wlp/bin/productInfo version` aus, um sicherzustellen, dass Ihre WLP-Version Bereitstellungen von Spring Boot 2.7.x mit `javaee-8.0` unterstützt. - **Java-Version**: Stellen Sie sicher, dass die von WLP verwendete Java-Version (überprüfen Sie mit `java -version` in der WLP-Umgebung) eine unterstützt, die von Spring Boot 2.7.18 unterstützt wird (Java 8, 11 oder 17).

Empfohlene Lösung

Basierend auf den bereitgestellten Informationen ist das wahrscheinlichste Problem ein Problem während der Initialisierung des Anwendungskontextes oder eine subtile Konfigurationsinkompatibilität mit WLP. Hier ist, wie Sie es beheben können:

1. Mit Protokollierung beginnen:

- Fügen Sie die `@PostConstruct`-Protokollierung (Schritt 2) und `debug=true` (Schritt 3) hinzu.
- Stellen Sie erneut bereit und überprüfen Sie `console.log` und `messages.log` auf Fehler oder wo der Start hängen bleibt.

2. Konfiguration überprüfen:

- Bestätigen Sie die WAR-Datei-Einrichtung und Abhängigkeiten (Schritt 4).
- Testen Sie lokal (Schritt 5), um Anwendungsprobleme auszuschließen.

3. Iterieren und vereinfachen:

- Wenn nötig, testen Sie ein minimales Projekt (Schritt 7), um das Problem zu isolieren.

Durch die Durchführung dieser Schritte sollten Sie die Ursache –sei es ein Abhängigkeitskonflikt, eine langsam startende Komponente oder ein WLP-Konfigurationsproblem –identifizieren und beheben können. Wenn Sie weiterhin Schwierigkeiten haben, teilen Sie spezifische Fehlermeldungen aus den Protokollen für weitere Unterstützung!