Konfiguration des Containers

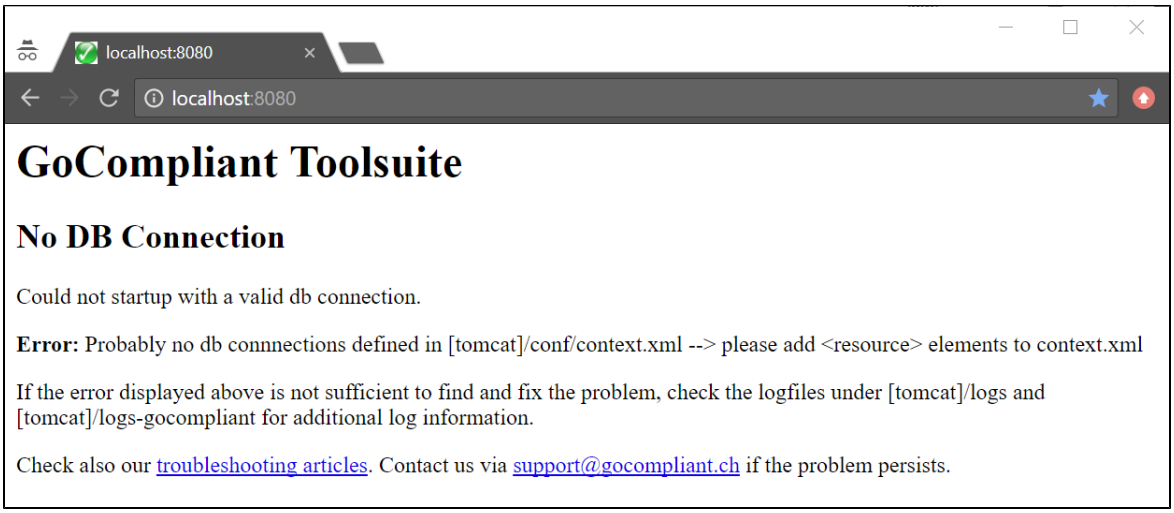
Testlauf

Testen Sie das Image, zum Beispiel mit "docker run --network host". Das Image öffnet per Default die Ports 8080 (HTTP), 8443 (HTTPS) und 8009 (AJP). Ersetzen Sie im folgenden Kommando das Tag 2.0.4-129 mit dem Tag des Ihnen vorliegenden Images, und führen Sie es aus:

Docker Run in Host Mode

```
docker run --network host gocompliant/toolsuite:2.0.4-129
```

Danach sollten Sie auf diesem Rechner mit einem Browser auf http://localhost:8080 zugreifen können und die folgende Fallback-Seite sehen, die bei fehlerhafter Datenbank-Konfiguration angezeigt wird.



Die Option "--network host" weist Docker an, den Network Stack des Hosts zu verwenden. Falls Sie dies nicht möchten oder die Ports 8080 / 8009 auf dem Host bereits belegt sind, führen Sie den Testlauf mit Port Mapping durch, z.B. auf den Port 8888:

Docker Run mit Portmapping

```
docker run -p 8888:8080 gocompliant/toolsuite:2.0.4-d-129
```

Importieren der Datenbank

Erstellen Sie Ihre Datenbankinstanz (MySQL oder Oracle) und importieren Sie den von uns zur Verfügung gestellten Datenbank-Dump. Der Dump enthält bereits die Benutzer "goco" und "quartz" sowie die zugehörigen Schemata und kann (Beispiel Oracle) mit folgendem Kommando importiert werden:

Import Datenbank-Dump

```
impdp ... schemas=goco,quartz directory=DUMP_DIR dumpfile=goco.dmp
```

Beispielkonfiguration mit docker-compose

Die Applikation benötigt Verbindungen zu der oben erwähnten Datenbank. Diese Verbindungen werden über die Tomcat-Konfiguration conf/context.xml in das Image injiziert. Erstellen Sie in einem beliebigen Verzeichnis ein File docker-compose.yml mit folgendem Beispiel-Inhalt:

Konfigurationsbeispiel mit docker-compose

```
version: '2'
services:
  gocompliant:
    image: gocompliant/toolsuite:2.0.4-d-129
    container_name: goco
    network_mode: host
    mem_limit: 3500m
    environment:
     - "JAVA_OPTS=-Xms2g -Xmx3g"
    volumes:
     - ./conf/context.xml:/usr/local/tomcat/conf/context.xml
     - ./logs:/usr/local/tomcat/logs
     - ./logs-gocompliant:/usr/local/tomcat/logs-gocompliant
```

Erstellen Sie danach das File ./conf/context.xml (bzw. passen Sie die Pfadangaben nach Ihren Vorstellungen an). Den Inhalt des Files können Sie dem Wiki-Abschnitt MySQL context.xml oder Oracle context.xml entnehmen. Statt des Networking Mode "host" können Sie natürlich auch "bridge" verwenden und den Datenbank-Listener sowie die Docker Network Bridge entsprechend konfigurieren, damit der Container eine Verbindung zu der ausserhalb des Containers laufenden Datenbank aufbauen kann.

Die beiden Log-Verzeichnisse des Containers werden in diesem Beispiel nach aussen in die Verzeichnisse ./logs (allgemeine Tomcat-Logs) und ./logs-gocompliant (spezifische GoCompliant-Logs) gemappt.

Optional: Installation ihrer Zertifikate

Tomcat ist für die Verwendung mit SSL (Port 8443) konfiguriert. Die Applikation bringt selbst-signierte Zertifikate mit, die durch Ihre eigenen ersetzt werden sollten. Der SSL-Connector ist im conf/server.xml bereits folgendermassen konfiguriert:

Vorkonfigurierter SSL Connector

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11AprProtocol"
                   maxThreads="150" SSLEnabled="true" >
        <!--<UpgradeProtocol className="org.apache.coyote.http2.Http2Protocol" />-->
        <SSLHostConfig>
                <Certificate certificateKeyFile="conf/goco-key.pem"
                                        certificateFile="conf/goco-cert.pem"
                                        certificateChainFile="conf/goco-certchain.pem" />
        </SSLHostConfig>
</Connector>
```

Wenn dieser Connector für Ihre Zwecke ausreicht, können Sie Ihre eigenen Zertifikate entsprechend benennen und über Volumes in /usr/local/tomcat /conf/ einbinden. Ansonsten erstellen Sie eine eigene conf/server.xml.

Start mit docker-compose

Nachdem Sie das docker-compose.yml erstellt haben, starten Sie das Image mit dem Befehl

Start mit docker-compose

```
docker-compose up -d
```

Sie können nun wieder mit dem Browser zugreifen und sollten eine Login-Seite sehen. Nehmen Sie mit uns Kontakt auf, um das initiale Login vorzunehmen und ein Authentisierungs-Verfahren zu konfigurieren.

Test-Instanz

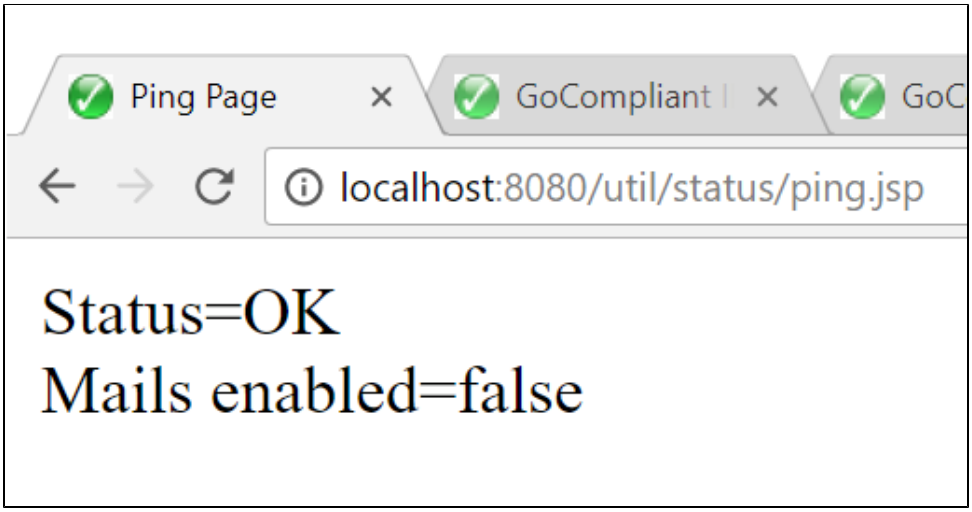
Sie können eine Test-Instanz als solche kennzeichnen, indem folgenden Parameter zu JAVA_OPTS hinzufügen: -Dgocompliant.environment=TEST. Diese Instanz wird dann in der Titelseite den Umgebungsnamen (TEST) anzeigen, sowie einen entsprechenden Zusatz zum Betreff von verschickten Emails hinzufügen. Ein Konfigurationsbeispiel:

Kennzeichnung als Test-Instanz

```
...
    environment:
     - "JAVA_OPTS=-Xms2g -Xmx3g -Dgocompliant.environment=TEST"
...
```

Monitoring

Sie können eine einfache Ping-Seite aufrufen unter /util/status/ping.jsp:



Verwandte Seiten

- Systemvoraussetzungen
- Systemparameter
- Informations- und Eskalations-Emails im IA
- Informations- und Eskalations-Emails im IKS
- Installation eines Updates