Continuous Integration con GitHub Actions

Table of Contents

Introduzione a GitHub Actions

Far riferimento alle seguenti risorse:

- https://docs.github.com/en/actions
- https://docs.github.com/en/actions/quickstart
- https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions

Scaricare il progetto

- Effettuare il fork del seguente progetto:
  - https://github.com/dduportal/dw-demo-app
- Scaricare Java JDK 8 (https://adoptium.net/temurin/archive/?version=8)
- Impostare la variabile JAVA_HOME alla nuova JDK
- Effettuare il checkout del progetto presente nel repository personale
- Provare ad eseguire il processo di build del progetto eseguendo:
  - `mvn install`

Esecuzione dell'applicazione

L'artefatto generato dall'applicazione è un `jar` standalone.

NOTE	`jar` = "Java Archive". E' un archivio zip che contiene le classi Java compilate e i metadata

Questo jar contiene al suo interno un *embedded application server*. Per questo l'esecuzione dell'applicazione non richiede un *Java application server* come Tomcat o Jetty

NOTE	Questo tipo di jar è conosciuto anche come "Uber-JAR"

L'applicazione verrà eseguita in 2 modi:

- **On the metal**: eseguendo l'embedded application server "as-is".
- **Tramite Docker**: build di una Docker image ed esecuzione di un Docker Container

Esecuzione "on the metal"

Andiamo ad eseguire l'applicazione "as is". Per un problema di porte non sarà possibile accedere all'applicazione da un web-browser, sarà comunque possibile verificare l'esecuzione da lina di comando.

IMPORTANT	L'obiettivo è di evidenziare com'è possibile eseguire dei test di Integrazione e degli Smoke test da un server senza interfaccia grafica.

- Eseguire il comando java con i seguenti parametri:
  - Path al jar file -jar
  - Il parametro "server" come primo argomento
  - Il path al file di configurazione YAML di DropWizard
- Il comando risultante è:

```
$JAVA_HOME/bin/java -jar ./target/demoapp.jar server ./hello-world.yml
```

- Una volta avviato verrà visualizzato il log nel terminale.
- Leggere le ultime linee di log:

```
...
INFO  [2016-08-31 16:38:12,271] org.eclipse.jetty.server.ServerConnector: Started
application@674658f7{HTTP/1.1}{0.0.0.0:8080}
INFO  [2016-08-31 16:38:12,272] org.eclipse.jetty.server.ServerConnector: Started
admin@5c8eee0f{HTTP/1.1}{0.0.0.0:8081}
INFO  [2016-08-31 16:38:12,272] org.eclipse.jetty.server.Server: Started @4018ms
```

- L'applicazione è attiva nella porta `8080`, e la consolle di amministrazione di DropWizard è attiva nella porta `8081`
- Eseguire il seguente comando:

```
curl -v http://localhost:8080/
```

Risultato atteso:

```
*   Trying ::1...
* Connected to localhost (::1) port 8080 (#0)
> GET / HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.50.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Wed, 31 Aug 2016 16:44:37 GMT
< Last-Modified: Wed, 31 Aug 2016 14:07:30 GMT
< Content-Type: text/html; charset=UTF-8
< Vary: Accept-Encoding
< ETag: "8f2181178186417ce600f7d5716bb124"
< Content-Length: 345
<
<!doctype html>

<head>
  <title>Hello Butler</title>
  <script src="angular.min.js"></script>
  <script src="hello.js"></script>
</head>



    <p>The ID is {{greeting.id}}</p>
    <p>The content is {{greeting.content}}</p>







* Connection #0 to host localhost left intact
```

- Ritornare al primo terminale. Verificare se viene visualizzato il seguente messaggio di log:

```
...
0:0:0:0:0:0:0:1 - - [31/Aug/2016:16:44:37 +0000] "GET / HTTP/1.1" 200 345 "-"
"curl/7.50.1" 5
...
```

- Terminare l'appicazione digitando la seguente combinazione di tasti: **CTRL-C**

```
...
^C
INFO  [2016-08-31 16:47:38,233] org.eclipse.jetty.server.ServerConnector: Stopped
application@674658f7{HTTP/1.1}{0.0.0.0:8080}
INFO  [2016-08-31 16:47:38,236] org.eclipse.jetty.server.ServerConnector: Stopped
```

```
admin@5c8eee0f{HTTP/1.1}{0.0.0.0:8081}
INFO  [2016-08-31 16:47:38,237] org.eclipse.jetty.server.handler.ContextHandler:
Stopped i.d.j.MutableServletContextHandler@66e889df{/,null,UNAVAILABLE}
INFO  [2016-08-31 16:47:38,239] org.eclipse.jetty.server.handler.ContextHandler:
Stopped i.d.j.MutableServletContextHandler@383790cf{/,null,UNAVAILABLE}
```

Continuous Integration con GitHub Actions

Di seguito vengono definiti i passi per configurare l'esecuzione del processo di Build e Test di un applicazione Java gestita con Maven con GitHub Actions

Configurare l'esecuzione del processo di Build in GitHub Actions

Seguire la guida presente a questo indirizzo: * https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-java-with-maven

GitHub Actions Workflow, Events, Jobs, Actions, Runners

Seguire la documentazione presente a questo indirizzo:

- https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions

Caratteristiche dell'ambiente di esecuzione del processo di build

Le caratteristiche degli ambienti dov'è possibile eseguire i job sono descrtte in questa pagina di documentazione.

Configurare il processo di build per il progetto di esempio

- Aprilre il seguente indirizzo: https://github.com/USER/PROJECT/actions/new
- Selezionare il link "Set up a workflow yourself"
- Creare il file `.github/workflows/build.yml` contenente il seguente contenuto:

```
name: Java CI with Maven

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v3
    - name: Set up JDK 1.8
```

```
      uses: actions/setup-java@v3
      with:
        distribution: 'temurin'
        java-version: 8.0.332+9
        cache: 'maven'
    - name: Build with Maven
      run: mvn -B install --file pom.xml
```

Il processo di build eseguirà in un ambiente con le seguenti caratteristiche:

- **Sistema Operativo**: Ubuntu 22.04
- Vengono eseguiti i seguenti steps:
  - Effettuato il checkout del codice grazie alla action https://github.com/actions/checkout
  - Configurata la jdk 8 grazie alla action https://github.com/actions/setup-java
  - Eseguito il comando `mvn -B install --file pom.xml`

Se tutto è andato a buon fine, all'indirizzo https://github.com/USER/PROJECT/actions/ sarà possibile vedere l'esecuzione e lo stato della build.

Per maggiori informazioni su come monitorare, risolvere problemi e attivare notifiche con Github actions consultare la seguente documentazione

Per maggiorni informazioni su come gestire informazioni sensibili consultare la seguente documentazione