a sample of karate project witch start the api to test before starting the test using docker compose

Sources

Testing Hasura GraphQL APIs with Karate
hasura · 1

karate/build-docker.sh at master - GitHub
github · 2

Spinning up Karat container - Stack
stackoverflow

Réponse

To start the API you want to test before running the Karate tests using Docker Compose, you can follow these steps:

1. Create a `docker-compose.yml` file that defines both the service for your API and the Karate test service.
2. In the `docker-compose.yml` file, define the API service first, so it starts before the Karate tests.
3. Add a `depends_on` directive in the Karate test service to ensure it waits for the API service to start.

Here's an example `docker-compose.yml` file:

```yaml
version: '3'
services:

  api:
    # Build or pull your API image
    image: your-api-image
    ports:
      - "8080:8080"

  karate-tests:
    image: karatelabs/karate
    depends_on:
      - api
    volumes:
      - ./tests:/tests
    working_dir: /tests
    command: ./run-tests.sh
```

In this example:

- The `api` service starts your API on port 8080.
- The `karate-tests` service uses the official Karate Docker image and mounts your tests directory as a volume.
- The `depends_on` directive ensures the `karate-tests` service waits for the `api` service to start.
- The `command` directive specifies a script (`run-tests.sh`) to run your Karate tests.

You'll need to create the `run-tests.sh` script in your `tests` directory to execute the Karate tests, e.g.:

```bash
#!/bin/bash
karate.jar classpath:karate-config.js classpath:tests/?*.feature
```

This script runs all `*.feature` files in the `tests` directory using the `karate-config.js` configuration file.

To run the tests, use the `docker-compose up` command in the same directory as your `docker-compose.yml` file:

```
docker-compose up
```

This will start the API service first, then the Karate tests will run against that API. [1] [4]

Partager

and how run the api automaticlay in my project before tests

Sources

Fully automated development environment with docker-compose
blog.maqpie · 1

Use Docker Compose
docs.docker · 2

Deploying with Do
api-platform ·

Réponse

To automatically run the API before running the tests in your project using Docker Compose, you can define the API service and the test service in the same `docker-compose.yml` file. The API service should be defined first, and the test service should have a `depends_on` directive to ensure it waits for the API service to start.

Here's an example `docker-compose.yml` file:

```yaml
version: '3'
services:

  api:
    build: ./api
    ports:
      - "8080:8080"

  tests:
    build: ./tests
    depends_on:
      - api
    command: ./run-tests.sh
    volumes:
      - ./tests:/tests
```

In this example:

- The `api` service builds and runs your API on port 8080.
- The `tests` service builds the test environment and runs the tests via the `run-tests.sh` script.
- The `depends_on` directive ensures the `tests` service waits for the `api` service to start before running the tests.
- The `volumes` directive mounts the `./tests` directory inside the `tests` container, allowing you to run tests from your local codebase.

You'll need to create the `run-tests.sh` script in your `tests` directory to execute the tests against the running API. For example, if you're using Karate for API testing, the script could look like this:

```bash
#!/bin/bash
karate.jar classpath:karate-config.js classpath:tests/?*.feature
```

This script runs all `*.feature` files in the `tests` directory using the `karate-config.js` configuration file.

To run the API and tests together, use the `docker-compose up` command:

```
docker-compose up
```

This will start the `api` service first, and once it's running, the `tests` service will automatically run the tests against the API. [1] [4]

Partager

can i integrate the start of compose up automatcly in pom or karate config or feature

Sources

Karate | Test Automation Made Simple.
karatelabs.github · 1

Configure a Karate Endpoint - TechDocs - Broadcom Inc.
techdocs.broadcom · 2

karate/build-dock
github · 3

Réponse

Yes, you can integrate the start of Docker Compose automatically in your Maven POM file or Karate configuration. Here are a few ways to achieve this:

Using Maven Plugin

You can use the Maven Docker Compose Plugin to start Docker Compose as part of your Maven build lifecycle. Add the plugin to your `pom.xml` file:

```xml
<plugin>
    <groupId>com.dkanejs.maven.plugins</groupId>
    <artifactId>docker-compose-maven-plugin</artifactId>
    <version>2.9.0</version>
    <executions>
        <execution>
            <id>start</id>
            <phase>pre-integration-test</phase>
            <goals>
                <goal>start</goal>
            </goals>
            <configuration>
                <detachedMode>true</detachedMode>
            </configuration>
        </execution>
    </executions>
</plugin>
```

This will start Docker Compose in detached mode before the `integration-test` phase, allowing your Karate tests to run against the running services. [1] [4]

Using Karate Configuration

You can configure Karate to start Docker Compose before running tests by adding a JavaScript function in your `karate-config.js` file:

```javascript
function() {
    var env = karate.env; // get java system property 'karate.env'
    karate.log('karate.env system property was:', env);
    if (!env) {
        env = 'dev'; // a intelligent default
    }
    var config = { // base config JSON
        appUrl: 'https://myapi.com'
    };

    // Start Docker Compose before tests
    if (env === 'e2e') {
        karate.exec('docker-compose up -d');
    }

    // don't waste time waiting for a service to start!
    karate.configure('connectTimeout', 5000);
    return config;
}
```

This code starts Docker Compose in detached mode if the `karate.env` system property is set to `e2e`. You can then run your Karate tests with `mvn test -Dkarate.env=e2e` to automatically start Docker Compose before the tests. [1] [5]

So in summary, you can either use a Maven plugin or configure Karate directly to start Docker Compose as part of your test execution workflow.

Partager

for the exemple on starting my compose with karate config my docker compose file is under resources/compose/docker-compose.yml

Réponse

To start the Docker Compose file located at `resources/compose/docker-c` your Karate configuration, you can modify the `karate-config.js` file as fo

```javascript
function() {
    var env = karate.env; // get java system property 'karate.env'
    karate.log('karate.env system property was:', env);
    if (!env) {
        env = 'dev'; // a intelligent default
    }
    var config = { // base config JSON
        appUrl: 'https://myapi.com'
    };

    // Start Docker Compose before tests
    if (env === 'e2e') {
        var composeFile = 'resources/compose/docker-compose.yml';
        var command = 'docker-compose -f ' + composeFile + ' up -d';
        karate.log('Running command:', command);
        karate.exec(command);
    }

    // don't waste time waiting for a service to start!
    karate.configure('connectTimeout', 5000);
    return config;
}
```

Here's what's happening:

1. The code checks if the `karate.env` system property is set to `e2e`.
2. If `karate.env` is `e2e`, it constructs the path to your Docker Compose file: `resources/compose/docker-compose.yml`.
3. It builds the `docker-compose` command with the `-f` flag to specify the file path, and the `up -d` flags to start the containers in detached mode.
4. The `karate.exec()` function is used to execute the constructed command, effectively starting the Docker Compose services.

To run your Karate tests and automatically start the Docker Compose services, you can use the following Maven command:

```
mvn test -Dkarate.env=e2e
```

This will set the `karate.env` system property to `e2e`, triggering the Doc
startup in your `karate-config.js` file before running the tests.

Make sure to replace `'https://myapi.com'` with the appropriate URL or configuration for your API under test.

Partager

can i configure env on application.yml for karate

Sources

How to work with Karate Config.js to Switch Env in Karate API ...
dev · 1

How to configure karate mock-servlet using a YAML file instead of ...
stackoverflow · 2

Step-2: Variables, Assertions - Kloia
kloia · 3

Réponse

Yes, you can configure the environment (env) in the `application.yml` file for Karate tests. Here's how you can do it:

1. In your `application.yml` file, define a property for the environment, for example:

```yaml
karate:
```