Profiles

Table of contents

With profiles you can define a set of active profiles so your Compose application model is adjusted for various usages and environments.

The services top-level element supports a `profiles` attribute to define a list of named profiles. Services without a `profiles` attribute are always enabled.

A service is ignored by Compose when none of the listed `profiles` match the active ones, unless the service is explicitly targeted by a command. In that case its profile is added to the set of active profiles.

> **Note**
>
> All other top-level elements are not affected by `profiles` and are always active.

References to other services (by `links`, `extends` or shared resource syntax `service:xxx`) do not automatically enable a component that would otherwise have been ignored by active profiles. Instead Compose returns an error.

Illustrative example

```
services:
 web:
   image: web_image

 test_lib:
   image: test_lib_image
   profiles:
     - test

 coverage_lib:
   image: coverage_lib_image
   depends_on:
     - test_lib
   profiles:
     - test

 debug_lib:
   image: debug_lib_image
   depends_on:
     - test_lib
   profiles:
     - debug
```

In the above example:

- If the Compose application model is parsed with no profile enabled, it only contains the `web` service.
- If the profile `test` is enabled, the model contains the services `test_lib` and `coverage_lib`, and service `web`, which is always enabled.
- If the profile `debug` is enabled, the model contains both `web` and `debug_lib` services, but not `test_lib` and `coverage_lib`, and as such the model is invalid regarding the `depends_on` constraint of `debug_lib`.
- If the profiles `debug` and `test` are enabled, the model contains all services; `web`, `test_lib`, `coverage_lib` and `debug_lib`.
- If Compose is executed with `test_lib` as the explicit service to run, `test_lib` and the `test` profile are active even if `test` profile is not enabled.
- If Compose is executed with `coverage_lib` as the explicit service to run, the service `coverage_lib` and the profile `test` are active and `test_lib` is pulled in by the `depends_on` constraint.
- If Compose is executed with `debug_lib` as the explicit service to run, again the model is invalid regarding the `depends_on` constraint of `debug_lib`, since `debug_lib` and `test_lib` have no common `profiles` listed.
- If Compose is executed with `debug_lib` as the explicit service to run and profile `test` is enabled, profile `debug` is automatically enabled and service `test_lib` is pulled in as a dependency starting both services `debug_lib` and `test_lib`.

See how you can use `profiles` in Docker Compose.