Docker Compose Experiment: Sync Files and Automatically Rebuild Services with Watch Mode

This article was fetched from an rss feed(20/04/2023)

We often hear how indispensable Docker Compose is as a development tool. Running `docker compose up` offers a streamlined experience and scales from quickly standing up a PostgreSQL database to building 50+ services across multiple platforms.

And, although “building a Docker image” was previously considered a last step in release pipelines, it’s safe to say that containers have since become an essential part of our daily workflow. Still, concerns around slow builds and developer experience have often been a barrier towards the adoption of containers for local development.

We’ve come a long way, though. For starters, Docker Compose v2 now has deep integration with BuildKit, so you can use features like RUN cache mounts, SSH Agent forwarding, and efficient COPY with –link to speed up and simplify your builds. We’re also constantly making quality-of-life tweaks like enhanced progress reporting and improving consistency across the Docker CLI ecosystem.

As a result, more developers are running `docker compose build && docker compose up` to keep their running development environment up-to-date as they make code changes. In some cases, you can even use bind mounts combined with a framework that supports hot reload to avoid the need for an image rebuild, but this approach often comes with its own set of caveats and limitations.

An early look at the watch command

Starting with Compose v2.17, we’re excited to share an early look at the new development-specific configuration in Compose YAML as well as an experimental file watch command (Figure 1) that will automatically update your running Compose services as you edit and save your code.

This preview is brought to you in no small part by Compose developer Nicolas De Loof (in addition to more than 10 bugfixes in this release alone).

![Screenshot of Docker compose command line showing the new "watch" command.](https://www.docker.com/wp-content/uploads/2023/04/Docker-watch-command-F1.gif "Screenshot of Docker compose command line showing the new "watch" command. - Docker watch command F1")

Figure 1: Preview of the new watch command.

An optional new section, `x-develop`, can be added to a Compose service to configure options specific to your project’s daily flow. In this release, the only available option is `watch`, which allows you to define file or directory paths to monitor on your computer and a corresponding action to take inside the service container.

Currently, there are two possible actions:

- `sync` — Copy changed files matching the pattern into the running service container(s).
- `rebuild` — Trigger an image build and recreate the running service container(s).

services: web: build: . x-develop: watch: - action: sync path: ./web target: /src/web - action: rebuild path: package.json

In the preceding example, whenever a source file in the `web/` directory is changed, Compose will copy the file to the corresponding location under `/src/web` inside the container. Because Webpack supports Hot Module Reload, the changes are automatically detected and applied.

Unlike source code files, adding a new dependency cannot be done on the fly, so whenever `package.json` is changed, Compose will rebuild the image and recreate the `web` service container.

Behind the scenes, the file watching code shares its core with Tilt. The intricacies and surprises of file watching have always been near and dear to the Tilt team’s hearts, and, as Dockhands, the geeking out has continued.

We are going to continue to build out the experience while gated behind the new `docker compose alpha` command and `x-develop` Compose YAML section. This approach will allow us to respond to community feedback early in the development process while still providing a clear path to stabilization as part of the Compose Spec.

Docker Compose powers countless workflows today, and its lightweight approach to containerized development is not going anywhere — it's just learning a few new tricks.

Try it out

Follow the instructions at dockersamples/avatars to quickly run a small demo app, as follows:

git clone https://github.com/dockersamples/avatars.git cd avatars docker compose up -d docker compose alpha watch

open http://localhost:5735 in your browser

If you try it out on your own project, you can comment on the proposed specification on GitHub issue #253 in the compose-spec repository.