Compose watch

Abolfazl Sabagh • Fri Nov 28 2025

Overview

Use Compose Watch

- Explanation
- Watch Path Rules

Compose Watch vs bind mounts

- Prerequisites

Configuration

- action
- path and target
- ignore
- initial_sync
- Example

Use Compose Watch

- Docker watch is compose attribute for all or only part of the project, might be suitable for automatic updates.
- Development hands-off tools, Automatically updates and previews your running Compose services as you edit and save your code.
- Compose Watch is designed to work with services built from local source code using the `build` attribute. not for services that rely on pre-built images specified by the `image` attribute.

Use Compose Watch

- **watch** adheres these file path rules:
  - All path relative to project directory apart from ignore pattern
  - Directories are watched recursively
  - Rules from `.dockerignore` apply
  - Use `ignore` option
  - Automatically ignored:
    - Temporary/backup files for common IDEs
    - `.git` directories
  - **Note:** Glob patterns aren't supported

Compose Watch versus bind mounts

- **Compose already supports** mounting host directories into service containers.
- **Watch mode is not a replacement** for this directory-sharing capability.
- **Watch mode is a complementary tool**, designed specifically to improve the development experience when working inside containers.
- More importantly, `watch` allows for **greater granularity** than is practical **with** a bind mount:
  - **Performance**
  - **Multi platform**

Configuration

- `watch` attribute defines a list of rules each rule requires a path pattern and action to take when a modification is detected
- **Prerequisites:**
  - `stat`
  - `mkdir`
  - `rmdir`

Configuration

- the container's USER can write to the target path so it can update files.

```
# Run as a non-privileged user
RUN useradd -ms /bin/sh -u 1001 app
USER app
# Install dependencies
COPY --chown=app:app . /app
```

Watch Rule: action

- **Sync :** If action is set to sync, Compose makes sure any changes made to files on your host automatically match with the corresponding files within the service container.
- Can be replacement for bind mount
- **Rebuild** : If action is set to rebuild, Compose automatically builds a new image and replaces the running service container.
- Equivalent : `docker compose up --build`
- Ideal for compiled languages or as a fallback for modifications to particular files that require a full image rebuild (e.g. package.json).
- **Optimize** your Dockerfile for with image layer caching and multi-stage builds.

Watch Rule: action

- **Sync + Restart :** compose sync your changes with the service containers and restarts them.
- Ideal when the config file changes, no need to rebuild. Just restart services
- `Initial_sync`**:** When using a sync+x action, the initial_sync attribute tells Compose to ensure files that are part of the defined path are up to date before starting a new watch session.

Watch Rule: path, target, ignore

```
myproject/
├── web/
│   ├── App.jsx
│   ├── index.js
│   └── node_modules/
├── Dockerfile
├── compose.yaml
└── package.json
```

```yaml
develop:
  watch:
    - action: sync
      path: ./web
      target: /src/web
      initial_sync: true
      ignore:
        - node_modules/
    - action: rebuild
      path: package.json
```

- `docker compose watch`
- `docker compose up --watch`