Docker Ofelia

docker, imagen

Sustituto de **cron** para contenedores, puede funcionar en modo fichero .ini o labels de docker. No los dos a la vez.

- https://hub.docker.com/r/mcuadros/ofelia
- https://pkg.go.dev/github.com/robfig/cron?utm_source=godoc

Parámetros config.ini

Parámetros de Control y Notificaciones (Válidos en ambos tipos)

Puedes añadir estos parámetros en cualquier sección para **sobrescribir los ajustes globales** solo para ese trabajo en concreto:

Parámetro	Descripción / Ejemplo
—	—
`email-to`	Envía los fallos de esta tarea a una dirección específica en lugar de la global. `email-to = dev-team@dominio.com`
`no-error-mail`	Si se establece en `true`, **desactiva** el envío de correos si esta tarea falla. `no-error-mail = true`
`slack-webhook`	Envía alertas de esta tarea a un Webhook de Slack concreto. `slack-webhook = [https://hooks.slack.com/](https://hooks.slack.com/)...`
`save-folder`	Ruta del host donde Ofelia guardará un archivo de texto con el log completo de la ejecución. `save-folder = /var/log/ofelia`
`overlap`	Controla si se permite ejecutar la tarea si la anterior ejecución **aún no ha terminado**. `overlap = false` *(por defecto `true`, si se pone `false` omite el inicio hasta que finalice el previo)*

job-run

`job-run` crea un contenedor nuevo a partir de una imagen, ejecuta el comando y lo destruye al terminar.

Parámetro	Requerido	Descripción / Ejemplo
—	—	—
`schedule`	**Sí**	Expresión Cron de 6 campos o sintaxis `@every`. `schedule = 0 0 1 * * *` o `@every 1h`
`image`	**Sí**	Nombre de la imagen de Docker a utilizar. `image = fidmag/python:3.5.3`
`command`	**Sí**	Comando o script que ejecutará la imagen. `command = python script.py`
`volume`	No	Rutas o volúmenes a montar. Se puede repetir para múltiples volúmenes. `volume = /app/host:/app:ro`
`environment`	No	Inyecta variables de entorno al contenedor. `environment = «VAR1=valor» «VAR2=123»`
`network`	No	Red de Docker a la que conectar el contenedor efímero. `network = mi_red_custom`
`user`	No	Usuario/UID con el que se ejecuta el proceso dentro del contenedor. `user = «1000:1000»` o `user = «www-data»`
`dir`	No	Directorio de trabajo (*working directory*) dentro del contenedor. `dir = /app`

`tty`	No	Asigna un pseudo-TTY (`true`/`false`). Útil si el comando requiere terminal. `tty = true`
`delete-on-completion`	No	Elimina el contenedor al terminar (equivale a `–rm`). Por defecto es `true`. `delete-on-completion = true`

job-exec

`job-exec` lanza un comando (`docker exec`) dentro de un contenedor que ya está activo en el sistema.

Parámetro	Requerido	Descripción / Ejemplo
—	—	—
`schedule`	**Sí**	Expresión Cron de 6 campos o sintaxis `@every`. `schedule = 0 30 * * * *`
`container`	**Sí**	Nombre exacto (`container_name`) o ID del contenedor destino. `container = mi_app_python`
`command`	**Sí**	Comando a ejecutar dentro del contenedor activo. `command = python /app/backup.py`
`user`	No	Usuario dentro del contenedor con el que ejecutar el comando. `user = «root»`
`dir`	No	Directorio desde el que se ejecutará el comando. `dir = /var/www/html`
`tty`	No	Asigna un pseudo-TTY para la ejecución (`true`/`false`). `tty = true`

From:
https://matebcn.eu/ - **miguel angel torres egea**

Permanent link:
https://matebcn.eu/tech:docker:ofelia

Last update: **26/08/2026 15:10**