Building with CMGR - Quick Start Guide

The Two Main Links You'll Need

- Start Problem Dev - Repo
- Install CMGR - Repo

Notes

- **CMGR overview:** CMGR is essentially a wrapper around Docker with extra features. The underlying containers are still Docker containers (you can view them with standard commands like `docker ps`).
- **Directory tree requirement:** CMGR expects all challenges to live inside its own project folder hierarchy (the "cmgr tree"), not scattered anywhere on your system. If a problem isn't in the right place in that hierarchy, CMGR won't find it.
    - (Quick check: if `cmgr update` doesn't find your problem, it's probably in the wrong location.)
- **More info:** See the two links above for full CMGR documentation.

CMGR Tip & Pitfalls to Avoid

- **Challenge location vs. namespace:** All challenges must be in the CMGR tree. However, when playtesting a challenge, the command uses the *namespace* + *challenge ID* (e.g., `picoctf/<challenge_id>`), not the directory path.
- **Flags and seeds:** Many CMGR challenges require flag and seed arguments. CMGR handles these automatically when you run problems through it. If you spin up a problem manually with Docker however (e.g., via docker build), you'll need to specify them with `--build-arg <ARG>=<VALUE>`.
- **Example problems:** The example-problems folder is very helpful.
    - `sanity-static-flag:` shows a basic challenge with downloadable artifacts (like a file)
    - `general-ssh:` shows a challenge where the user connects directly to the box
- **Publishing ports:**
    - If your Dockerfile includes a `#PUBLISH <port> AS <name>` line, you *must* reference that port name in your problem.md, or CMGR will complain. (See `general-ssh` for a working example.)
    - You only need to publish ports if the user must directly access the box or a hosted web service. CMGR automatically manages the port for the problem description page.
    - Two key notes about publishing:

        - `PUBLISH … AS` is **not** a Docker directive; it's a CMGR tag, so keep it commented.
        - The directive must be capitalized (`PUBLISH … AS`), or CMGR won't recognize it.

A Good CMGR Problem Workflow (IMO)

1. Identify the type of problem:
    - Downloadable artifacts only
    - Web challenge
    - Direct access to a box
2. Find a similar example problem and copy its Dockerfile and problem.md.
3. Modify the Dockerfile as needed and test with Docker first (Docker gives much more verbose error output than CMGR).
4. Once it works correctly in Docker, test it with CMGR.
5. Before submitting the challenge, make sure of a few things:
    - The correct flag is accepted and wrong flags are rejected
    - The flag is accessible (aka play test your problem)
    - The problem.md has an appropriate name, description, details, etc., and has been cleaned up
    - There is a solve script (feel free to look at / modify solve scripts from other challenges)