Use OpenCode with Docker Model Runner

- Prerequisites
- Step 1: Pull a coding model
- Step 2: Create an OpenCode configuration
- Step 3: Verify the endpoint
- Step 4: Start OpenCode
- Step 5: Package gpt-oss with a larger context window
- Troubleshooting
- Learn more

This guide shows how to connect OpenCode to Docker Model Runner so OpenCode can use local models for coding tasks. You'll configure an `opencode.json` file, verify the API endpoint, and run OpenCode against models served from your local Docker environment.

> **Acknowledgment**
>
> Docker would like to thank Pradumna Saraf for his contribution to this guide.

Unlike the OpenCode sandbox guide, this guide focuses on using OpenCode as a local coding tool backed by Docker Model Runner rather than running OpenCode in a containerized sandbox.

In this guide, you'll learn how to:

- Pull coding models for OpenCode
- Configure OpenCode to use Docker Model Runner
- Verify the local API endpoint and start OpenCode
- Package `gpt-oss` with a larger context window when you need it

Prerequisites

Before you start, make sure you have:

- Docker Desktop or Docker Engine installed
- Docker Model Runner enabled
- OpenCode installed

If you use Docker Desktop, turn on TCP access in **Settings** > **AI**, or run:

```
$ docker desktop enable model-runner --tcp 12434
```

Step 1: Pull a coding model

Pull one or more models before you configure OpenCode:

```
$ docker model pull ai/qwen3-coder
$ docker model pull ai/devstral-small-2
```

These models are a good fit for coding workflows because they support large context windows.

Step 2: Create an OpenCode configuration

OpenCode reads configuration from either of these locations:

- `~/.config/opencode/opencode.json` for a global setup
- `opencode.json` in your project root for a project-specific setup

Project-level configuration overrides the global file.

Add a provider that points to Docker Model Runner:

```
{
  "$schema": "https://opencode.ai/config.json",
  "provider": {
```

```
    "dmr": {
      "npm": "@ai-sdk/openai-compatible",
      "name": "Docker Model Runner",
      "options": {
        "baseURL": "http://localhost:12434/v1"
      },
      "models": {
        "qwen3-coder": {
          "name": "ai/qwen3-coder"
        },
        "devstral-small-2": {
          "name": "ai/devstral-small-2"
        }
      }
    }
  }
}
```

This configuration adds Docker Model Runner as an OpenCode provider and exposes two local models.

> [!NOTE]
>
> If your setup expects the older OpenAI-compatible path, use `http://localhost:12434/engines/v1` instead.

Step 3: Verify the endpoint

Check that Docker Model Runner is reachable before you open OpenCode:

```
$ curl http://localhost:12434/v1/models
```

If you use the older path, run:

```
$ curl http://localhost:12434/engines/v1/models
```

The response should list the models available through Docker Model Runner.

Step 4: Start OpenCode

From your project directory, run:

```
$ opencode
```

To switch models from the TUI, run:

```
/models
```

Then select the model from the `dmr` provider.

Step 5: Package `gpt-oss` with a larger context window

This step is optional. Use it if you need a larger context window for repository-scale tasks.

`gpt-oss` defaults to a smaller context window than coding-focused models. If you want to use it for repository-scale tasks, package a larger variant:

```
$ docker model pull ai/gpt-oss
$ docker model package --from ai/gpt-oss --context-size 128000 gpt-oss:128k
```

Then add it to your OpenCode configuration:

```
{
  "$schema": "https://opencode.ai/config.json",
  "provider": {
    "dmr": {
      "npm": "@ai-sdk/openai-compatible",
      "name": "Docker Model Runner",
      "options": {
        "baseURL": "http://localhost:12434/v1"
      },
      "models": {
        "gpt-oss:128k": {
```

```
          "name": "gpt-oss:128k"
        }
      }
    }
  }
}
```

Troubleshooting

If OpenCode can't connect, check Docker Model Runner status:

```
$ docker model status
```

If OpenCode does not show your model, list local models:

```
$ docker model ls
```

If the model is missing, pull it first and confirm the model name in `opencode.json` matches the local model you want to use.

Learn more

- Docker Model Runner overview
- Docker Model Runner API reference
- IDE and tool integrations

□□□ #2
□ □□□ □□□ 2026-10-31 18:47:09 UTC
□ □□□ □□□ 2026-10-31 18:47:10 UTC