@host.today/ht-docker-tools

contains docker images for specific tools

readme.md for @host.today/ht-docker-tools

□□ ht-docker-tools

Production-ready Docker images for MongoDB, MinIO, ClickHouse, Caddy, and Redis — supercharged with Deno and Node.js runtimes.

These images let you run your favorite infrastructure tools alongside modern JavaScript/TypeScript tooling. Whether you need to script database migrations, automate object storage, or run custom health checks — you're covered.

Issue Reporting and Security

For reporting bugs, issues, or security vulnerabilities, please visit community.foss.global/. This is the central community hub for all issue reporting. Developers who sign and comply with our contribution agreement and go through identification can also get a code.foss.global/ account to submit Pull Requests directly.

□□ Why ht-docker-tools?

- **Scripting-Ready**: Every image includes Deno for running TypeScript scripts directly in containers
- **Two Flavors**: Choose official base images (minimal changes) or Alpine images (lightweight with full Node.js/NVM support)
- **Multi-Arch**: Full support for `linux/amd64` and `linux/arm64` (Apple Silicon friendly)
- **CI/CD Optimized**: NVM works out of the box in GitHub Actions, GitLab CI, and any `docker exec bash -c` workflow

---

□□ Available Images

Official Base Images

Extend official Docker images with Deno runtime. Ideal for production with existing configurations.

Image	Base	What's Added
`ht-docker-tools:clickhouse`	`clickhouse/clickhouse-server`	+ Deno
`ht-docker-tools:mongodb`	`mongo:latest`	+ Deno
`ht-docker-tools:minio`	`minio/minio:latest`	+ Deno
`ht-docker-tools:caddy`	`caddy:latest`	+ Deno
`ht-docker-tools:redis`	`redis:latest`	+ Deno

Alpine Images ⚡

Lightweight Alpine-based images with the full stack. Perfect for CI/CD and development.

Image	Includes	Approx. Size
`ht-docker-tools:clickhouse-alpine`	ClickHouse + NVM + Node.js 20 + Deno	~250MB
`ht-docker-tools:mongodb-alpine`	MongoDB + NVM + Node.js 20 + Deno	~200MB
`ht-docker-tools:minio-alpine`	MinIO + NVM + Node.js 20 + Deno	~200MB
`ht-docker-tools:caddy-alpine`	Caddy + NVM + Node.js 20 + Deno	~200MB
`ht-docker-tools:redis-alpine`	Redis + NVM + Node.js 20 + Deno	~200MB

□□ Quick Start

Pull from Registry

```
# Official base images
docker pull code.foss.global/host.today/ht-docker-tools:mongodb
```

```
docker pull code.foss.global/host.today/ht-docker-tools:redis

# Alpine images (recommended for most use cases)
docker pull code.foss.global/host.today/ht-docker-tools:mongodb-alpine
docker pull code.foss.global/host.today/ht-docker-tools:redis-alpine
```

Run with Deno Scripts

```
# Start MongoDB
docker run -d --name mongo code.foss.global/host.today/ht-docker-tools:mongodb

# Execute a Deno script inside the container
docker exec mongo deno run --allow-net --allow-read script.ts
```

Use Node.js with NVM (Alpine Images)

```
# Start Redis Alpine
docker run -d --name redis code.foss.global/host.today/ht-docker-tools:redis-alpine

# Check Node version
docker exec redis bash -c "node --version"
# v20.18.2

# Switch Node versions on the fly
docker exec redis bash -c "nvm install 18 && nvm use 18 && node --version"
# v18.x.x
```

□□ Docker Compose Example

```
services:
  mongodb:
    image: code.foss.global/host.today/ht-docker-tools:mongodb-alpine
    ports:
      - "27017:27017"
    volumes:
```

```
      - mongodb_data:/data/db
    command: mongod

  redis:
    image: code.foss.global/host.today/ht-docker-tools:redis-alpine
    ports:
      - "6379:6379"
    volumes:
      - redis_data:/data
    command: redis-server

  minio:
    image: code.foss.global/host.today/ht-docker-tools:minio-alpine
    ports:
      - "9000:9000"
      - "9001:9001"
    environment:
      MINIO_ROOT_USER: minioadmin
      MINIO_ROOT_PASSWORD: minioadmin
    volumes:
      - minio_data:/data
    command: minio server /data --console-address ":9001"

  caddy:
    image: code.foss.global/host.today/ht-docker-tools:caddy-alpine
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./Caddyfile:/etc/caddy/Caddyfile
      - caddy_data:/data
      - caddy_config:/config

volumes:
  mongodb_data:
  redis_data:
  minio_data:
  caddy_data:
  caddy_config:
```

□□ NVM in Docker — It Just Works™

NVM is notoriously tricky in Docker because it's a bash function, not a binary. These Alpine images solve this with a smart entrypoint that handles three contexts:

Context	How It Works
Dockerfile RUN	Custom shell wrapper loads NVM before each command
CI/CD `bash -c`	Entrypoint detects and injects bashrc sourcing
Interactive shells	Standard bashrc initialization

```
# Works in CI/CD pipelines
docker exec mycontainer bash -c "nvm use 18 && npm install && npm test"

# Works in Dockerfiles
FROM code.foss.global/host.today/ht-docker-tools:mongodb-alpine
RUN nvm install 18 && npm install -g typescript
```

□□ Building Locally

```
# Clone the repository
git clone https://code.foss.global/host.today/ht-docker-tools.git
cd ht-docker-tools

# Build all images (native architecture)
./build-images.sh

# Test all images
./test-images.sh
```

Multi-Architecture Builds

For CI/CD pipelines pushing to registries:

```
docker buildx build \
  --platform linux/amd64,linux/arm64 \
  -f Dockerfile_mongodb_alpine \
  -t code.foss.global/host.today/ht-docker-tools:mongodb-alpine \
  --push .
```

□□ Bundled Versions

Component	Version
NVM	v0.40.1
Node.js LTS	v20.18.2
Deno	Latest (from Alpine edge/community)

Alpine images use `unofficial-builds.nodejs.org` for musl-compatible Node.js binaries.

□□ Use Cases

- **Database Migrations**: Run TypeScript migration scripts directly in your DB container
- **Health Checks**: Write sophisticated health probes in Deno
- **Scripted Backups**: Automate backup tasks with Node.js scripts
- **CI/CD Integration**: Execute tests that need both database and runtime
- **Local Development**: Single container with everything you need

---

License and Legal Information

Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH and any related products or services are trademarks of Task Venture Capital GmbH and are not included within the scope of the MIT license granted herein. Use of these trademarks must comply with Task Venture Capital GmbH's Trademark Guidelines, and any usage must be approved in writing by Task Venture Capital GmbH.

Issue Reporting and Security

For reporting bugs, issues, or security vulnerabilities, please visit community.foss.global/. This is the central community hub for all issue reporting. Developers who sign and comply with our contribution agreement and go through identification can also get a code.foss.global/ account to submit Pull Requests directly.

Company Information

Task Venture Capital GmbH Registered at District court Bremen HRB 35230 HB, Germany

For any legal inquiries or if you require further information, please contact us via email at hello@task.vc.

By using this repository, you acknowledge that you have read this section, agree to comply with its terms, and understand that the licensing of the code does not imply endorsement by Task Venture Capital GmbH of any derivative works.