

CHAP-DEVOPS

MLproject models with chap-devops

Serve it with no build, evaluate it, and test it with chap-core

For people who have a model in the MLproject format

An MLproject and a chapkit service

	MLproject model	chapkit service
What it is	a directory with an <code>MLproject</code> file and scripts	a container with an HTTP API
How it runs	chap runs the <code>train</code> and <code>predict</code> commands	chap calls it over HTTP
An image to build	not with <code>chap-devops chapkit mlproject run</code>	yes, from its <code>Dockerfile</code>
In a deployment	registers while <code>mlproject run</code> serves it	a service that chap-devops manages

`chap-devops chapkit mlproject run` makes the first kind act as the second, with no image to build. You need Docker. Python, uv, R and chapkit are not necessary.

Versions

- `chap-devops chapkit mlproject run` uses chapkit images of its own. It works with chap-core v2.4.0 and later.
- `mlproject export , --run-config.as-chapkit` and the built-in models need chap-core v2.5.0, the newest release.
- In a deployment on an older tag, `chap-devops update` moves it to v2.5.0.

Make or bring the model

A model from chap-models, or your own, works as it is. For a new one:

```
chap-devops models new rain-model --template mlproject-r  
cd rain-model && git init && git add . && git commit -m "feat: first version"  
cd ..
```

```
created rain-model in rain-model (mlproject-r template, service id rain-model)  
next: run `chap-devops chapkit mlproject run rain-model` to serve it with no build,  
or build the image as `rain-model/README.md` says
```

For Python, use `--template mlproject-py`.

Serve it, with no build

```
chap-devops chapkit mlproject run rain-model
```

```
serving /home/me/rain-model with `chapkit mlproject run`  
  in ghcr.io/dhis2-chap/chapkit-r-tidyverse-cli:latest  
  (from the train command `Rscript scripts/train.R ...`; --image picks another)  
starting it on http://localhost:5001; the logs follow, and Ctrl-C stops it  
the MLproject service answers at http://localhost:5001
```

It worked when the last line says the MLproject service answers at .

In the foreground, until Ctrl-C

- The service runs in the foreground, and the logs of chapkit follow.
- Use a second terminal for the next steps.
- Ctrl-C stops the service and removes its container:

```
stopped the MLproject service on http://localhost:5001
```

- chap-devops mounts the model directory at `/work`.
- The port is the first free port from 5001.
- The options after `DIR` go to chapkit, such as `--log-level debug`.

Which image: `chap-devops chapkit --image`

NAME	PLATFORMS	WHAT IT HAS
<code>py</code>	amd64, arm64	Python
<code>py-data</code>	amd64, arm64	Python, numpy, pandas, scipy, scikit-learn, statsmodels, ...
<code>r</code>	amd64, arm64	R, renv, pak
<code>r-tidyverse</code>	amd64, arm64	R, tidyverse, fable, tsibble, forecast, xgboost, ...
<code>r-inla</code>	amd64	R, tidyverse, INLA, sf, spdep, dlnm, ...

Each name is `ghcr.io/dhis2-chap/chapkit-NAME-cli:latest`. The real output also has a `LOCAL` column: whether this docker has the image already.

The default image, and `--image`

The train command in `MLproject` selects the image:

The train command	The image
<code>python</code> or <code>uv</code>	<code>py-data</code>
<code>Rscript</code>	<code>r-tidyverse</code>

```
chap-devops chapkit --image r-inla mlproject run my-model  
chap-devops chapkit --image my-model-base:dev mlproject run my-model
```

- A model that needs INLA needs `--image r-inla`.
- An image of your own must have chapkit 2.4.0 or later. Build it `FROM` a `-cli` image.

Packages: what chap-devops does not install

- chap-devops does not install the packages that `uv_env` or `renv_env` names.
- The image has only the packages of the table.
- If the model needs another package, the train job stops with an error such as `No module named 'cyclopts'`.

Then do one of these:

1. Give `--image` with an image that has the package.
2. Let chap install the packages: `chap-devops chap --run-config.as-chapkit`.

Evaluate it

In a second terminal, while the service runs:

```
chap-devops chap eval --model-name http://host.docker.internal:5001 \  
  --dataset-csv laos_subset.csv --output-file rain-model.nc
```

```
running `chap eval` in ghcr.io/dhis2-chap/chap-core:master  
files: chap reads and writes in /home/me  
model: http://host.docker.internal:5001 answers  
chap finished; it wrote rain-model.nc  
`chap-devops chap plot-backtest rain-model.nc --output-file rain-model.html` plots it
```

Why not `localhost`

- `chap-devops chap` runs `chap` in a container.
- In the container, `localhost` is the container itself. A `localhost` URL reaches nothing there.
- `host.docker.internal` is the name of this machine in the container.
- So give the port that `chap-devops` printed, at `host.docker.internal`.

`chap-devops` refuses a `localhost` URL before the run, and gives the same port at `host.docker.internal`.

Or let chap run the directory itself

Without a service, chap runs the `train` and `predict` commands in its worker image:

```
chap-devops chap eval --model-name ./rain-model \  
  --dataset-csv laos_subset.csv --output-file rain-model.nc
```

With `--run-config.as-chapkit`, chap serves the directory with chapkit in its container, and installs what `uv_env` or `renv_env` names:

```
chap-devops chap eval --model-name ./rain-model --run-config.as-chapkit \  
  --dataset-csv laos_subset.csv --output-file rain-model.nc
```

In a deployment: it registers with chap-core

```
chap-devops init lab --models none
cd lab
chap-devops up --wait
chap-devops chapkit mlproject run ../rain-model
```

```
the MLproject service answers at http://localhost:5001; it registers with
  chap-core in this deployment, which reaches it at http://chap-devops-mlproject-18970-0:8000
```

The commands of the deployment use the model by its id: the `name` in `MLproject`.

In a deployment: configure it and test it

In a second terminal, in `lab`:

```
$ chap-devops models configs sync rain-model
rain-model: created configured model default
$ chap-devops models test rain-model --backtest
testing 1 model (through chap-core: a dataset, a backtest and its scores)
rain-model    pass      13s   crps 31.4   mae 31.4   rmse 37.5
  configured model: default (id 21)

1 of 1 model passes
```

With DHIS2 in the deployment, the Modeling App shows the configured model while the service runs.

Commit first: chap-core needs a git revision

- chap-core stores a model only with a git revision.
- chap-devops gives chapkit the git commit of the model directory.
- If the directory has changes that are not committed, chap-devops warns.

After a change:

1. Commit the change.
2. Raise `version` in `MLproject`.
3. Run `chap-devops chapkit mlproject run DIR` again.

`chap-devops status` lists it as `unmanaged`, because chap-devops does not manage it.

In the Modeling App

With DHIS2 in the deployment, the served model is in **Select model**, by the `name` in `MLproject`, with the commit that chap-devops gave chapkit.

- An evaluation runs as for any other model.
- `rain-model` has no covariates, so map only **Disease cases**.
- Tested: Modeling App 7.3.0, DHIS2 2.42.6, chap-core v2.5.0.
- 18 provinces of Laos, 2019 to 2024, 7 splits: 19 seconds.
- chap-core also ran its three baselines on the same data.

Make it a chapkit service: `mlproject migrate`

Commit first: the migration moves files to `_old/`. In the model directory:

```
$ chap-devops chapkit mlproject migrate . --yes
running `chapkit mlproject migrate` in ghcr.io/dhis2-chap/chap-core:v2.5.1
chapkit finished; it wrote .dockerignore, .gitignore, CHAPKIT.md, Dockerfile,
  Makefile, README.md, _old/.dockerignore, _old/.gitignore, and 6 more
wrote uv.lock with `uv lock` in the same image, so `docker build` works
```

chapkit finds the scripts in `scripts/` through the train and predict commands. chap-core v2.5.0 and older need `--base-image`.

Build it, run it, test it

```
docker build --platform linux/amd64 \
  --build-arg GIT_REVISION=$(git rev-parse HEAD) -t rain-model:dev .
```

```
$ chap-devops run rain-model:dev
running rain_model on http://localhost:5001 (answered in 2s)
$ chap-devops -C ~/.local/share/chap-devops/run/default models test rain_model
1 of 1 model passes
```

It worked when the last line says **1 of 1 model passes**.

And back: `mlproject export`

A chapkit service that runs scripts can write the `MLproject` of the same model:

```
$ chap-devops chapkit mlproject export main.py -o MLproject
running `chapkit mlproject export` in ghcr.io/dhis2-chap/chap-core:master
chapkit finished; it wrote MLproject
```

- Only a service that runs scripts, such as one from `shell-py` or `shell-r`.
- The exported `MLproject` runs only with chapkit's runner:
`chap-devops chapkit mlproject run`, or `--run-config.as-chapkit`.
- It does not run with `chap-devops chap` alone.

What was tested

On chap-core `master` before v2.5.0 (chapkit 2.4.0), macOS arm64. Each model was served with `mlproject run` and evaluated with `chap-devops chap eval`:

Image	The models that pass
<code>py-data</code> (default)	<code>mlproject-py</code> <code>template</code> , <code>minimalist_configurable_model</code> , <code>minimal_template_example</code> , <code>Xiang_SVM</code>
<code>r-tidyverse</code> (default)	<code>mlproject-r</code> <code>template</code> , <code>mean</code> , <code>XGBoost_for_Malawi</code>
the default	<code>shell-py</code> and <code>shell-r</code> templates, exported
<code>--image r-inla</code>	<code>INLA_baseline_model</code> , <code>ewars_per_district</code>

What was tested: the models that fail

All three ran in `py-data`, the default image:

Model	Result
<code>chap_pymc</code>	fails: the image has no <code>cyclopts</code> , which its <code>uv_env</code> names
<code>IMPA-TECH-SARIMA</code>	fails in the model, also without chap-devops
<code>rwanda_random_forest</code>	fails in its train script

A package that the image does not have: see **Packages: what chap-devops does not install.**

When something fails: serve and test

chap-devops or chap says	What to do
is not a git checkout, so chap-core cannot store the model	git init, git add ., git commit, then serve again
has changes that are not committed	Commit, raise version in MLproject, serve again
No module named '...'	Give --image, or use --run-config.as-chapkit
`http://localhost:...` is this machine	Use http://host.docker.internal:PORT
an MLproject service has no `chapkit test`	Use --backtest, or migrate the model
stopped, because its container got SIGTERM	Something stopped the container; run the same command again

When something fails: migrate and export

chap-devops or chapkit says	What to do
No Python (.py) or R (.r/.R) scripts	Give <code>--base-image</code> , such as <code>chapkit-py</code>
Only ShellModelRunner services can be exported	Move the model to scripts, as in <code>shell-py</code>
No such file or directory: 'config.yml'	Add <code>--run-config.as-chapkit</code> , or use <code>mlproject run</code>
could not write DIR/uv.lock	Run the migration again on a new copy; <code>uv lock</code> needs the network

Each message is in the book's Troubleshooting chapter, with more detail.

Start now

```
chap-devops models new rain-model --template mlproject-r
cd rain-model && git init && git add . && git commit -m "feat: first version"
chap-devops chapkit mlproject run .
```

- The guide, step by step:
<https://winterop-com.github.io/chap-devops/use-cases/mlproject-model.html>
- The chapter about `chap-devops chapkit`:
<https://winterop-com.github.io/chap-devops/chapkit-cli.html>