AWS SAM CLI Serverless on AWS

START HERE

`sam init`
New project

`sam deploy --guided`
First deploy, saves config

`sam build`
Build into .aws-sam

`sam sync --stack-name <name> --watch`
Live-sync changes

`sam local invoke`
Run one function in Docker

`sam logs --stack-name <name> --tail`
Follow Lambda logs

`sam local start-api`
Local API on :3000

`sam delete`
Remove the stack

template.yaml SAM template → `sam build` → **.aws-sam/build** built artifacts → `sam deploy` → **CloudFormation** change set → `creates` → **Lambda, API, ...** per stack

CONCEPTS

A SAM template is CloudFormation with shorthand resources (AWS::Serverless::Function). sam build prepares code, sam deploy hands the template to CloudFormation.

Local commands (local invoke, start-api) need Docker running. Run sam build first so they use the built code.

sam deploy --guided writes samconfig.toml; later runs of sam deploy need no flags.

PROJECT

`sam init`
Create a project interactively

`sam init --runtime <runtime> --name <name> --app-template <template> --no-interactive`
Create a project without prompts
Zip package type; use --package-type Image with --base-image for containers.

`sam validate`
Check the template is valid

`sam validate --lint`
Also lint with cfn-lint

`sam list resources --stack-name <name>`
Show resources and physical IDs

`sam list endpoints --stack-name <name>`
Show API and function endpoints

`sam list stack-outputs --stack-name <name>`
Show stack outputs

BUILD

`sam build`
Build all functions and layers
Output goes to .aws-sam/build; deploy and local commands pick it up.

`sam build <FunctionId>`
Build a single resource

`sam build --use-container`
Build in a Lambda-like container
Use for natively compiled dependencies.

BUILD (cont.)

`sam build --cached --parallel`
Reuse unchanged builds, build in parallel

LOCAL TESTING

`sam local invoke`
Run the only function once

`sam local invoke <FunctionId> --event <file>`
Run a function with an event file

`sam local invoke --env-vars <file>`
Set environment variables from JSON or .env

`sam local invoke --debug-port <port>`
Run with a debugger port exposed

`sam local start-api`
Serve the API on localhost:3000

`sam local start-api --port <port>`
Listen on another port

`sam local start-api --warm-containers EAGER`
Keep containers between requests
LAZY starts each container on first call.

`sam local start-lambda`
Emulate the Lambda endpoint locally

`sam local generate-event`
List services with sample events

`sam local generate-event <service> <event>`
Print a sample event
Redirect to a file, then pass it with --event.

DEPLOY

`sam deploy --guided`
Answer prompts, save samconfig.toml

`sam deploy`
Deploy using saved config

`sam deploy --stack-name <name> --resolve-s3 --capabilities CAPABILITY_IAM`
Deploy with no prompts
CAPABILITY_NAMED_IAM if roles have custom names.

DEPLOY (cont.)

`sam deploy --no-confirm-changeset`
Skip the changeset prompt

`sam deploy --no-execute-changeset`
Create the changeset only

`sam deploy --parameter-overrides ParameterKey=<k>,ParameterValue=<v>`
Set a template parameter

`sam deploy --config-env <env>`
Use a named samconfig.toml section

`sam deploy --no-fail-on-empty-changeset`
Exit 0 when nothing changed
By default an empty changeset exits non-zero.

`sam deploy --profile <profile> --region <region>`
Pick credentials and region

`sam deploy --output json`
Machine-readable output for CI
Not compatible with --guided or --confirm-changeset.

SYNC & REMOTE

`sam sync --stack-name <name> --watch`
Auto-sync code and infra changes
Dev stacks only: code updates go straight to Lambda via service APIs.

`sam sync --stack-name <name> --code`
Push code changes once, skip CloudFormation

`sam sync --stack-name <name> --code --resource-id <FunctionId>`
Sync a single resource

`sam remote invoke <FunctionId> --stack-name <name>`
Invoke the deployed function

`sam remote invoke <FunctionId> --stack-name <name> --event '<json>'`
Send an inline event

`sam remote invoke <FunctionId> --stack-name <name> --event-file <file>`
Send an event from a file

LOGS & TRACES

`sam logs --stack-name <name>`
Fetch recent logs for the stack
Defaults to the last 10 minutes.

LOGS & TRACES (cont.)

`sam logs --stack-name <name> --tail`
Follow logs live

`sam logs -n <FunctionId> --stack-name <name>`
Logs for one function

`sam logs --stack-name <name> --filter "<pattern>"`
Only matching log events

`sam logs --stack-name <name> -s '<time>' -e '<time>'`
Logs in a time range
Accepts '10min ago' or '2018-01-01 10:10:10'.

`sam logs --stack-name <name> --include-traces`
Add X-Ray traces to the output

`sam traces --tail`
Follow X-Ray traces

CONFIG & CLEANUP

`sam delete`
Delete the stack (prompts)

`sam delete --stack-name <name> --no-prompts`
Delete without asking
Cannot be undone; check the stack name first.

`sam <command> --debug`
Debug logging with timestamps

`sam <command> --help`
Show a command's options

RECIPES

First deploy

```
sam init
cd <name>
sam build
sam deploy --guided
```

Test a function locally

```
sam local generate-event s3 put > event.json
sam build
sam local invoke <FunctionId> -e event.json
```

Fast dev loop

```
sam sync --stack-name <name> --watch
# edit code: it syncs on save
sam logs --stack-name <name> --tail
```

CI deploy

```
sam validate --lint
sam build --use-container
sam deploy --no-confirm-changeset --no-fail-on-empty-changeset
```