:doctype: book

include::attributes.txt[]

// Attributes

[.topic] [#build-containers] = Build and deploy container image assets in CDK apps :info\_titleabbrev: Build and deploy container image assets :keywords: Docker, Dockerfile, Finch, Container, Container image, Container image asset, Amazon ECR, {aws} CDK, {aws} CDK CLI, Deploy, Build

== [abstract]

## When you build container image assets with the {aws} Cloud Development Kit ({aws} CDK), Docker is utilized by default to perform these actions. If you want to use a different container management tool, you can replace Docker through the CDK\_DOCKER environment variable.

// Content start

When you build container image assets with the {aws} Cloud Development Kit ({aws} CDK), Docker is utilized by default to perform these actions. If you want to use a different container management tool, you can replace Docker through the CDK\_DOCKER environment variable.

[#build-containers-intro-example] == Example: Build and publish a container image asset with the {aws} CDK

The following is a simple example of an {aws} CDK app that builds and publishes a container asset to Amazon Elastic Container Registry (Amazon ECR) using Docker by default:

*Project structure*:: + [source,none] — my-cdk-app/ ├── lib/ │ ├── my-stack.ts │ └── docker/ │ ├── Dockerfile │ └── app/ │ └── index.js ├── bin/ │ └── my-cdk-app.ts ├── package.json ├── tsconfig.json └── cdk.json —

*Dockerfile*:: + [source,auto] — FROM public.ecr.aws/lambda/nodejs:16

= Copy application code

COPY app/ /var/task/

= (Optional) Install dependencies

= RUN npm install

= The Lambda Node.js base image looks for index.handler by default

’’’

*Application code*:: + In lib/docker/app/index.js: + [source,javascript] — console.log(“Hello from inside the container!”); —

*CDK stack*:: + [source,javascript] — import \* as cdk from ‘aws-cdk-lib’; import { Construct } from ‘constructs’; import \* as ecr\_assets from ‘aws-cdk-lib/aws-ecr-assets’;

export class MyStack extends cdk.Stack { constructor(scope: Construct, id: string) { super(scope, id);

…. // Define a Docker image asset const dockerImageAsset = new ecr\_assets.DockerImageAsset(this, ‘MyDockerImage’, { directory: ‘lib/docker’, // Path to the directory containing the Dockerfile });

// Output the ECR URI new cdk.CfnOutput(this, ‘ECRImageUri’, { value: dockerImageAsset.imageUri, }); } } —- ….

*CDK app*:: + [source,javascript] — #!/usr/bin/env node import \* as cdk from ‘aws-cdk-lib’; import { MyStack } from ‘../lib/my-stack’;

const app = new cdk.App(); new MyStack(app, ‘MyStack’); —

When we run cdk deploy, the {aws} Cloud Development Kit ({aws} CDK) Command Line Interface (CLI) does the following:

. *Build the Docker image* – Run docker build locally based on the Dockerfile in the specified directory (lib/docker). . *Tag the image* – Run docker tag to tag the built image with a unique hash, based on the image contents. . *Publish to Amazon ECR* – Run docker push to publish the container image to an Amazon ECR repository. This repository must already exist. It is created during the default bootstrapping process. . *Output the Image URI* – After a successful deployment, the Amazon ECR URI of the published container image is output in your command prompt. This is the URI of our Docker image in Amazon ECR.

[#build-container-replace] == How to replace Docker with another container management tool

Use the CDK\_DOCKER environment variable to specify the path to the binary of your replacement container management tool. The following is an example of replacing Docker with [noloc]Finch:

== [source,none,subs=“verbatim,attributes”]

$ which finch /usr/local/bin/finch # Locate the path to the binary

$ export CDK\_DOCKER=‘/usr/local/bin/finch’ # Set the environment variable

== $ cdk deploy # Deploy using the replacement

Aliasing or linking is not supported. To replace Docker, you must use the CDK\_DOCKER environment variable.

[#build-container-supported] == Supported Docker drop-in replacement engines

[noloc]Finch is supported, although there may be some Docker features that are unavailable or may work differently as the tool evolves. For more information on Finch, see https://aws.amazon.com/blogs/opensource/ready-for-flight-announcing-finch-1-0-ga/[Ready for Flight: Announcing Finch 1.0 GA!] in the *{aws} Open Source Blog*.

Other container management tools may work. The CDK doesn’t check which Docker replacement you are using to determine if it’s supported. If the tool has equivalent Docker commands and behaves similarly, it should work.