Abstracting IaC on AWS

A fundamental principle of DevOps is to treat infrastructure the same way developers treat code. Infrastructure provisioning, orchestration, and deployment should also support the use of the infrastructure as code (IaC). AWS provides a number of IaC offerings.

One offering is AWS CloudFormation, which is a service you can use to specify any cloud infrastructure you need in a simple template file. CloudFormation then provisions the infrastructure for you. Developers can use CloudFormation to create AWS resources in an orderly and predictable fashion. Depending on the complexity of the architecture you want to create—especially if you are creating many serverless resources—CloudFormation templates can be lengthy. For example, a CloudFormation template might need many lines, which could make it more difficult to reuse when you want to share code with the DevOps team. To help manage this complexity, AWS created the AWS Cloud Development Kit (AWS CDK).

The AWS CDK is an abstraction layer that provides a framework for designing cloud infrastructure in code using a language of your choice. It will then build a CloudFormation template out of the infrastructure you define. You can use the AWS CDK to model application infrastructure with TypeScript, Python, Java, and .NET. Developers can use their existing integrated development environment (IDE) and take advantage of tools like autocomplete and inline documentation to accelerate infrastructure development. The AWS Serverless Application Model (AWS SAM) and the AWS CDK both abstract IaC to make deployments easier. (The previous course covered AWS SAM. If you're not familiar with the topic, you can refer to the previous readings.)

By generating your CloudFormation templates from source code, you can extend the integration with external sources. You can use toolkits and resources that are available for the supported programming languages—like variables, data structures, external API calls, loops, functions, arrays, and other—and do any tasks that are possible with a programming language. For example, say you want to create five Amazon Elastic Compute Cloud (Amazon EC2) instances. If you used only CloudFormation, you would need to copy and paste five individual resources of the type `AWS::EC2::Instance` into the template. However, if you use Python with the AWS CDK, you can create the five EC2 instances by putting a construct that renders the instances inside a `for` loop, which is not possible by using only CloudFormation.

Benefits of using the AWS CDK

- **Easier cloud onboarding -** The AWS CDK accelerates your onboarding to AWS because there are fewer new things to learn (after you already know the basics of the programming language you decided to use).
- **Faster development process -** The AWS CDK gives you the expressive power of programming languages for defining infrastructure.

- **Customizable and shareable -** With the AWS CDK, you can design your own reusable components that meet your organization’s requirements for security, compliance, and governance.
- **No context switching -** You can build your cloud application with the AWS CDK without leaving your IDE or command line interface (CLI).

The AWS CDK Toolkit is the primary tool for interacting with your AWS CDK application. In the AWS Command Line Interface (AWS CLI), use the `cdk` command. The AWS CDK Toolkit runs your application, checks the application model you defined, and produces and deploys the CloudFormation templates that were rendered by the AWS CDK.

Resources

Introduction to DevOps on AWS: Infrastructure as Code
AWS CDK Developer Guide
What Is CDK? - Introduction to CDK (workshop.aws)
AWS CDK Intro Workshop | AWS Cloud Development Kit (AWS CDK) Workshop (cdkworkshop.com)