Automating Deployment Using CI/CD Pipelines

DEVOPS · CI/CD · AWS CODE SERVICES · CLOUDFORMATION · AWS SAM

STUDY & REVIEW

01 MUST KNOW

if you remember five things, remember these

01 **Your app is app + infrastructure:** in the cloud your application is not just your code — it is the code **plus all of the associated infrastructure** (VPCs, Elastic Load Balancing, Auto Scaling groups, RDS, S3, ElastiCache, and more). **DevOps** is the combination of cultural philosophies, practices, and tools that supports building the two together; its motto is *"People over process over tools."*

02 **CI → delivery → deployment: continuous integration (CI)** merges every change into a central repo and verifies it with an automated build and test — commonly thought to **stop at the build stage**. **Continuous delivery** extends CI out to production-like testing but keeps a **manual intervention** step before release; **continuous deployment** automates release to customers all the way through, with **no human intervention**.

03 **Three AWS code services: CodeBuild** compiles source, runs tests, and produces deployable packages (CI); **CodeDeploy** automates deployments to EC2, Fargate, Lambda, and on-premises servers; **CodePipeline** orchestrates the build, test, and deploy phases into an automated release pipeline (CD) on every code change.

04 **CloudFormation = infrastructure as code:** a fully managed service that provisions groups of related resources from **JSON- or YAML-formatted templates** into **stacks** (a stack is a *unit of deployment*). **Resources** is the only required template section, and deleting a stack **deletes all of its resources** by default.

05 **AWS SAM for serverless:** an open-source framework and **extension of CloudFormation** for serverless apps. The `Transform: AWS::Serverless-2016-10-31` declaration is **required** to mark a template as SAM; you deploy the SAM template **with CloudFormation**, and it integrates with CodeBuild, CodeDeploy, and CodePipeline.

02 KEY TERMS

TERM	DEFINITION
DevOps	Combination of Development and Operations — the cultural philosophies, practices, and tools that support cloud application development and automate the development and release process
DevSecOps	DevOps taken one step further by integrating security into the same combination of philosophies, practices, and tools
Continuous integration (CI)	Regularly merging code changes into a central repository and verifying each change with an automated build and test; commonly thought to stop at the build stage
Continuous delivery	Extends CI by testing out to production-like stages, keeping a manual intervention step between check-in and release to customers
Continuous deployment	Fully automated release of software to customers — from check-in through production with no human intervention
Infrastructure as code (IaC)	Automating the creation, update, and deletion of AWS infrastructure with repeatable code, so identical environments come from one code base
Microservice architecture	Writing an application as discrete services that run independently in containers or as Lambda functions, instead of one monolithic service
AWS CodeBuild	Fully managed continuous integration service that compiles source code, runs tests, and produces deployable packages — no build servers to manage
AWS CodeDeploy	Fully managed deployment service that automates software deployments to EC2, Fargate, Lambda, and on-premises servers
AWS CodePipeline	Fully managed continuous delivery service that automates the build, test, and deploy phases of a release pipeline each time code changes

TERM	DEFINITION
Stack	A CloudFormation unit of deployment — the collection of related resources generated from a template; deleting it deletes those resources
AWS SAM	Serverless Application Model — an open-source, CloudFormation-based framework for defining, locally testing, and deploying serverless applications

03 AWS CODE SERVICES FOR CI/CD

one-line job of each — exam gold

SERVICE	CI/CD ROLE	WHAT IT DOES
AWS CodeBuild	Continuous integration	Compiles source code, runs tests, and produces packages ready to deploy; scales continuously and runs concurrent builds — no build servers to provision
AWS CodeDeploy	Deployment	Automates deployments to EC2, Fargate, Lambda, and on-premises servers; avoids downtime and eliminates error-prone manual steps
AWS CodePipeline	Continuous delivery	Automates build, test, and deploy on every code change, per the release model you define; integrates with GitHub and custom plugins, pay for what you use

04 CONTINUOUS DELIVERY VS. CONTINUOUS DEPLOYMENT

the difference is one manual gate

CONTINUOUS DELIVERY

manual gate before customers

- Extends CI out to production-like stages with verification testing
- May reach all the way to production deployment
- Keeps a **manual intervention** step between check-in and release
- Goal: greater certainty the software works in production

CONTINUOUS DEPLOYMENT

fully automated to production

- Automated release of software straight to customers
- Runs from check-in through production — **no human intervention**
- Shrinks time to deliver value and to get customer feedback
- Enables a fast, iterative customer feedback loop

05 A CI/CD PIPELINE

CodePipeline orchestrates every stage on each code change

`01` Source — change merged to the central repo → `02` Build — **CodeBuild** compiles and packages → `03` Test — automated tests verify the change → `04` Manual intervention — optional approval gate → `05` Deploy — **CodeDeploy** releases to EC2 / Fargate / Lambda

06 CLOUDFORMATION TEMPLATE SECTIONS

only Resources is required

SECTION	PURPOSE
AWSTemplateFormatVersion	Template format version; `2010-09-09` is the latest and only valid value
Description	A text string that describes the template
Metadata	Objects that provide additional information about the template
Parameters	Values passed in at runtime when you create or update a stack
Mappings	A key-to-value lookup table, e.g. for conditional parameter values
Conditions	Control whether resources are created or properties set (e.g. prod vs. test)
Transform	For serverless (Lambda) apps — specifies the AWS SAM version to use
Resources	The stack resources and their properties — **the only required section**
Outputs	The values returned when you view the stack's properties

07 EXAM TRAPS

where the knowledge check gets you

- × “Continuous delivery and continuous deployment mean the same thing.” — Continuous **delivery** keeps a manual intervention step before customers see a change; continuous **deployment** automates all the way to production with **no human intervention**.
- × “Continuous integration includes releasing to production.” — CI is commonly thought to **stop at the build stage**: it merges changes to a central repo and runs an automated build and test. Releasing is delivery or deployment.
- × “CloudFormation templates have to be written in JSON.” — Templates can be **JSON or YAML** text files; CloudFormation follows the ECMA-404 JSON standard and the YAML 1.1 spec, with a few exceptions.
- × “Every section of a CloudFormation template is required.” — Only the **Resources** section is required; format version, parameters, outputs, and the rest are optional.
- × “Any date works for AWSTemplateFormatVersion.” — `2010-09-09` is the **latest and only valid** template format version.
- × “Deleting a stack leaves its resources running.” — By default, deleting a stack **deletes all of its resources** (the behavior can be reconfigured and overridden).
- × “CloudFormation stack quotas are fixed and can't be raised.” — Request an increase through the **Support Center in the AWS Management Console** for quotas without a fixed maximum.
- × “AWS SAM is a separate deployment engine from CloudFormation.” — SAM is an **extension of CloudFormation**: you deploy a SAM template *with* CloudFormation, and the required `Transform: AWS::Serverless-2016-10-31` declaration marks it as SAM.

08 SELF-QUIZ

answers are upside-down below

Q1 DevOps is a combination of which two functions, plus what three kinds of things?

Q2 What is the DevOps cultural motto about priorities?

Q3 What does DevSecOps add to DevOps?

Q4 At which stage is continuous integration commonly thought to stop?

Q5 What single difference separates continuous delivery from continuous deployment?

Q6 Which AWS code service compiles source code, runs tests, and produces deployable packages?

Q7 Which AWS code service automates deployments to EC2, Fargate, Lambda, and on-premises servers?

Q8 Which AWS code service automates the build, test, and deploy phases into a release pipeline?

Q9 Which is the only required section of a CloudFormation template?

Q10 Which transform declaration identifies a template as an AWS SAM template?

ANSWER KEY (rotate the page)

Q1 Development and Operations — a combination of cultural philosophies, practices, and tools **Q2** “People over process over tools” **Q3** It integrates security into the DevOps philosophies, practices, and tools **Q4** The build stage **Q5** Delivery keeps a manual intervention step before release; deployment is fully automated with no human intervention **Q6** AWS CodeBuild **Q7** AWS CodeDeploy **Q8** AWS CodePipeline **Q9** The Resources section **Q10** Transform: AWS::Serverless-2016-10-31