

# AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

D O P 3 0 8 - R

# Develop AWS CloudFormation templates to manage your infrastructure

Matteo Rinaudo (he/him)

Sr. Developer Advocate  
Amazon Web Services – AWS CloudFormation

Dan Blanco (he/him)

Sr. Developer Advocate  
Amazon Web Services – AWS CloudFormation



# Agenda

- Introduction to workshops
- CloudFormation overview
- Local development tools
- Template architecture
- Hands-on lab
- Review

# Introduction



# Workshop anatomy: Theory and practice

## Theory

- Learn from slides
- Review docs

## Practice

- Learn by doing
- Hands-on lab

# AWS Workshop Studio



<https://catalog.workshops.aws/>



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# AWS Workshop Studio

## AWS Workshop Studio

join hands on events and  
workshops

Find physical and virtual self-paced Workshops, GameDays, Bootcamps, Immersion Days, and other events that require hands-on access to AWS accounts.

### Join an event

All you need is an event access code to join and get started with your event

[Get started](#)



# AWS Workshop Studio

## Sign in

Choose a preferred sign-in method

**Email one-time password (OTP)**

Enter your personal or corporate email to receive a one-time password

**Login with Amazon**

Login with your Amazon.com retail account

**Amazon employee**

Login with your Amazon Corporate account. Only for Amazon Employees.





# AWS Workshop Studio

## One-time email passcode

We sent a passcode to  
receive it within 5 minutes.

You should

Passcode (9-digit) [Resend passcode](#)

Back

Sign in

[Get help signing in](#)

# AWS Workshop Studio

Workshop Studio > Join event

Step 1

Enter event access code

## Enter event access code

Step 2

Review and join

**Event access code**

Event access code

A 12 digit code that was given to you for this event

*abcd-012345-ef*

Cancel

Next

Access code: **eede-006813-cc**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# AWS Workshop Studio

[Workshop Studio](#) > [Join event](#)

Step 1

[Enter event access code](#)

Step 2

**Review and join**

## Review and join

### Event details

Name	Start time	Duration	Level
Architecting your templates - AWS CloudFormation Workshop	9/27/2022 10:53 AM	72 hours	-

### Description

This is a test-only event for the Architecting your templates workshop lab.



# CloudFormation overview



# CloudFormation background

- Announced on February 25, 2011
- Model, provision, and manage AWS and third-party resources
- Scale your infrastructure worldwide
- Manage resources across AWS accounts and regions with a single operation
- Extend CloudFormation
  - Include cloud resources published in the CloudFormation registry
  - Extension available today: Modules, resource types, hooks

# CloudFormation fundamentals

Template – describe your infrastructure as code

- Written in declarative YAML/JSON
- CloudFormation-specific DSL
- Source code for your cloud architecture

Stack – manage your cloud infrastructure

- Create, update, and delete resources
- Import/drift detection
- Change sets for dry runs

# CloudFormation fundamentals

Template – describe your infrastructure as code

- Written in declarative YAML/JSON
- CloudFormation-specific DSL
- Source code for your cloud architecture

Stack – manage your cloud infrastructure

- Create, update, and delete resources
- Import/drift detection
- Change sets for dry runs

# CloudFormation templates

---

AWSTemplateFormatVersion: "version date"

Description:

string

Metadata:

template metadata

Parameters:

set of parameters

Rules:

set of rules

Mappings:

set of mappings

Conditions:

set of conditions

Transform:

set of transforms

Resources:

set of resources

Outputs:

set of outputs





# CloudFormation templates

---

**AWSTemplateFormatVersion:** "version date"

**Description:**

string

**Metadata:**

template metadata

**Parameters:**

set of parameters

**Rules:**

set of rules

**Mappings:**

set of mappings

**Conditions:**

set of conditions

**Transform:**

set of transforms

**Resources:**

set of resources

**Outputs:**

set of outputs



# Parameters

- Describe template parameters (max 200) in the Parameters section
- Can be of type String, Number, List<Number>, CommaDelimitedList
- You can specify
  - Allowed patterns
  - Allowed values
  - Min/max length
  - Min/max value
- You can specify a default value for a parameter (constraints you describe apply)

# Parameters example

Parameters:

InstanceTypeParameter:

Type: String

Default: t2.micro

AllowedValues:

- t2.micro
- m1.small
- m1.large

Description: Enter t2.micro, m1.small, or m1.large.

# Resources

- Describe resources (max 500) in the required Resources section
- Specify
  - The Logical ID (A-Za-z0-9) – must be unique in the template
  - Use the Logical ID to reference the resource in the template
  - The resource type (e.g., `AWS::EC2::Instance`) and resource properties
- Update behaviors: depending on which resource property you change
  - No interruption (e.g., Tags for `AWS::CloudTrail::Trail`)
  - Some interruption (e.g., InstanceType for `AWS::EC2::Instance`)
  - Replacement (e.g., ImageId for `AWS::EC2::Instance`)

# Resources example

Resources:

Ec2Instance:

Type: AWS::EC2::Instance

Properties:

InstanceType: **!Ref** 'InstanceTypeParameter'

ImageId: ami-0123abcd

BlockDeviceMappings:

- DeviceName: /dev/sdm

Ebs:

VolumeType: io1

# Resources example

Resources:

Ec2Instance:

Type: AWS::EC2::Instance

Properties:

**InstanceType: !Ref 'InstanceTypeParameter'**

ImageId: ami-0123abcd

BlockDeviceMappings:

- DeviceName: /dev/sdm

Ebs:

VolumeType: io1

# Intrinsic functions

- CloudFormation's built-in functions
  - Built-in functions to the CloudFormation template language
  - Longform (**Fn::GetAtt**) or shortform (**!GetAtt**) [YAML]
  - Processed as template is consumed
  - Resolve values
- Examples
  - **!Ref**, **!GetAtt** – resolve values from resources, or parameters
  - **!Sub** – string manipulation
  - **!ImportValue** – get values from other stacks
  - **!Split**, **!Join**, **!Length**, **!Select** – array manipulation

And more!



# Pseudo parameters

- CloudFormation's built-in variables
  - Prefixed with **AWS::**
  - Filled in automatically by CloudFormation
  - Allow easier modularization
  - Avoid hard-coding
- Examples
  - **AWS::StackName** – name of the stack (useful for uniqueness)
  - **AWS::Region** – Region stack is deployed in (useful for cross-Region)
  - **AWS::AccountId** – account stack is deployed in (useful for AWS IAM Identity Center [successor to AWS Single Sign-On])

And more!



# Outputs

- Describe outputs (max 200) in the Outputs section
- Declare outputs to
  - Easily find values relevant to resources you described in the template
  - Export values in the current account and region

Outputs:

InstanceId:

Description: The ID of the EC2 instance.

Value: **!Ref** 'Ec2Instance'

# Local development



# Use a code editor/IDE

- Visual Studio Code
- Sublime Text
- Atom
- IntelliJ IDEA
- PyCharm
- Vim
- Emacs
- More!

And more!

# CloudFormation Linter

```
1  AWSTemplateFormatVersion: "2010-09-09"
2  Description: A sample template
3  • Errors:
4    Catch: Missing
5  Parameters:
6  • myParam:
7    Type: String
8    Default: String
9    Description: String
10 Resources:
11   ## Missing Properties
12   • MyEC2Instance1:
13     Type: "AWS::EC2::Instance1"
14     ## Fake Properties Key on main level
15     ## Bad sub properties in BlockDeviceMappings/Ebs and NetworkInterfaces
16     MyEC2Instance:
17       Type: "AWS::EC2::Instance"
18   • Properties:
19     ImageId: "ami-2f726546"
20     InstanceType: t1.micro
21     KeyName: 1
22     FakeKey: MadeYouLook
23     BlockDeviceMappings:
24     -
```

## Validate CloudFormation templates

- Best practices
- Resource specification
- Value-checking
- CLI-enabled
- IDE plugins

Severity	Provider	Description	Line
Warning	Cfn-Lint	Top level item Errors isn't valid	3:1
Warning	Cfn-Lint	Parameter myParam not used	6:1
Warning	Cfn-Lint	Invalid Type AWS::EC2::Instance1 for resource MyEC2Instance1	12:1
Warning	Cfn-Lint	Properties not defined for resource MyEC2Instance1	12:1
Warning	Cfn-Lint	Invalid Property FakeKey for resource MyEC2Instance	18:1
Warning	Cfn-Lint	Invalid Property BadSubX2Key for resource MyEC2Instance	26:1



# CloudFormation Guard



```
~/dev/guard
> cat template.yaml | cfn-guard validate --rules sse.guard
Other
STDIN Status = FAIL
FAILED rules
sse.guard/dynamo_db_sse_on_for_prod_only          FAIL
---
Evaluating data STDIN against rules sse.guard
Number of non-compliant resources 1
Resource = DDBShouldFail {
  Type      = AWS::DynamoDB::Table
  Rule = dynamo_db_sse_on_for_prod_only {
    ALL {
      Check = Properties.SSESpecification.SSEType EQUALS %allowed_algorithms {
        RequiredPropertyError {
          PropertyPath = /Resources/DDBShouldFail/Properties/SSESpecification[L:17,C:18]
          MissingProperty = SSEType
          Reason = Could not find key SSEType inside struct at path /Resources/DDBShouldFail/Properties/SSESpecification[L:17,C:18]
          Code:
            15.      Value: AppSelected
            16.      SSESpecification:
            17.      SSEEnabled: true
        }
      }
    }
  }
  Check = DeletionPolicy EQUALS "Retain" {
    RequiredPropertyError {
      PropertyPath = /Resources/DDBShouldFail[L:11,C:8]
      MissingProperty = DeletionPolicy
      Reason = Could not find key DeletionPolicy inside struct at path /Resources/DDBShouldFail[L:11,C:8]
      Code:
        9.      SSEEnabled: true
        10.     DDBShouldFail:
```

## Policy-as-code tool for IaC

- Write rules and validate templates
- PASS/FAIL conditions
- Unit-testable
- Rulegen for existing templates
- CLI-enabled

# CloudFormation fundamentals

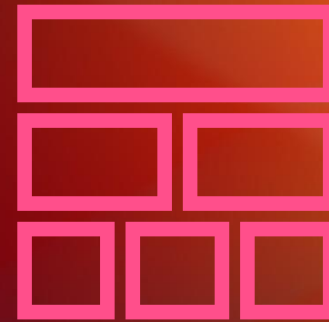
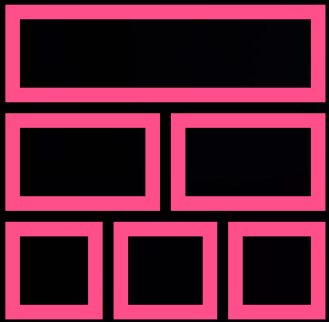
Template – describe your infrastructure as code

- Written in declarative YAML/JSON
- CloudFormation-specific DSL
- Source code for your cloud architecture

Stack – manage your cloud infrastructure

- Create, update, and delete resources
- Import/drift detection
- Change sets for dry runs

# Stacks



# Stacks





# Template architecture



# Conceptualizing your architecture

**Frontend  
resources**

Instances, AWS Auto Scaling groups

**Stateful  
resources**

Databases and clusters, queues

**Backend  
services**

API endpoints, functions

**Monitoring  
resources**

Alarms, dashboards

**Base  
network**

VPCs, NAT gateways, VPNs, subnets

**Identity &  
security**

IAM users, groups, roles, policies

1

Break stacks by  
ownership and  
lifecycle

2

Reuse stacks by  
environments

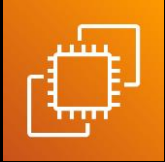
Dev

Test

Stage

Prod

# Sample application



Amazon EC2



Amazon Aurora



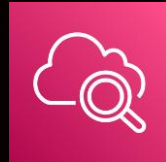
Load Balancer



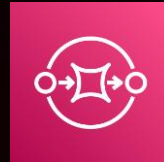
Auto Scaling



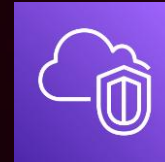
Amazon S3



CloudWatch

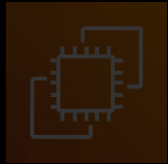


Amazon SQS



Amazon VPC

# Stateful resources



Amazon EC2



Amazon Aurora



Load Balancer



Auto Scaling



Amazon S3



CloudWatch

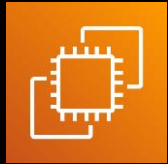


Amazon SQS



Amazon VPC

# Backend services



Amazon EC2



Amazon Aurora



Load Balancer



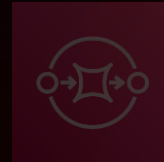
Auto Scaling



Amazon S3



CloudWatch

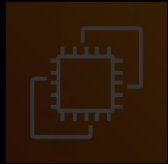


Amazon SQS



Amazon VPC

# Networking resource



Amazon EC2



Amazon Aurora



Load Balancer



Auto Scaling



Amazon S3



CloudWatch

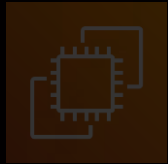


Amazon SQS



Amazon VPC

# Monitoring resource



Amazon EC2



Amazon Aurora



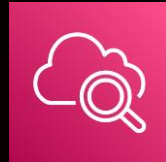
Load Balancer



Auto Scaling



Amazon S3



CloudWatch



Amazon SQS



Amazon VPC

# Ownership matters!



Amazon EC2



Amazon Aurora



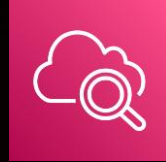
Load Balancer



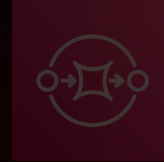
Auto Scaling



Amazon S3



CloudWatch



Amazon SQS



Amazon VPC



# Hands-on lab



# AWS Workshop Studio

Architecting your templates - AWS CloudFormation Workshop

AWS CloudFormation Workshop

▶ Introduction

▶ Prerequisites

▶ Basics

▼ Intermediate

▼ Templates

Conditions

Resource Dependencies

Dynamic References

Nested stacks

Layered stacks

Package and deploy

Architecting your templates

Policy-as-code with Guard

▼ AWS account access

[Open AWS console](#)

[Get AWS CLI credentials](#)


Exit event

Event in progress

Ends in 2 days 23 hours 44 minutes 16 seconds.

Event dashboard > AWS CloudFormation Workshop

## AWS CloudFormation Workshop



### Welcome to the AWS CloudFormation Workshop!

The intent of this workshop is to educate builders about the features of [AWS CloudFormation](#) and how to get started building quickly. A background in CloudFormation, command line, git, and development workflows is not required.

Previous

Next

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# AWS Workshop Studio

## AWS CloudFormation Workshop

- ▶ Introduction
- ▶ Prerequisites
- ▶ Basics
- ▼ Intermediate
  - ▼ Templates
    - Conditions
    - Resource Dependencies
    - Dynamic References
    - Nested stacks
    - Layered stacks
    - Package and deploy
    - Architecting your templates**
    - Policy-as-code with Guard

# AWS Workshop Studio

## ▼ AWS account access

[Open AWS console](#) 

[Get AWS CLI credentials](#)

# Stuck? Raise your hand!

We're friendly



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Didn't finish?



<https://catalog.workshops.aws/cfn101/en-US>

# Recap



# Summary

- Leverage a good editor and tools, like CloudFormation Linter
- Dig into the CloudFormation template anatomy
- Understand intrinsic functions and pseudo parameters
- Keep your stacks small and modular
- Architect for lifecycle and ownership



# Thank you!

Matteo Rinaudo

 @mrinaudo

Mastodon:

@mrinaudo@awscommunity.social

Dan Blanco

 @TheDanBlanco

Mastodon:

@danblanco@awscommunity.social



Please complete the session survey in the **mobile app**

