AWS Solution Architect

Monitoring and Automation

A Day in the Life of an AWS Administrator

Mr. William works for an IT company as a Solution architect. The company has several cloud-based applications. Recently, some of its servers crashed due to high workload, and the company rebuilt its infrastructure with high server capacity servers. This consumed a significant amount of time, and the company incurred a huge loss in that.

Now, the company wants Mr. William to assure that these types of failures should not occur in the future. While searching for a service that can help him in performance monitoring of the resources and quick replication of the infrastructure, he found that CloudWatch and CloudFormation are two services offered by AWS that can solve his problem.

In this lesson, he will get a brief idea about CloudWatch and CloudFormation.

Learning Objectives

By the end of this lesson, you will be able to:

- Monitor the instances using CloudWatch alarms
- Create CloudFormation template
- Update resources in an existing infrastructure
- Create an infrastructure using CloudFromation
- Implement the best practices of CloudFormation

CloudWatch

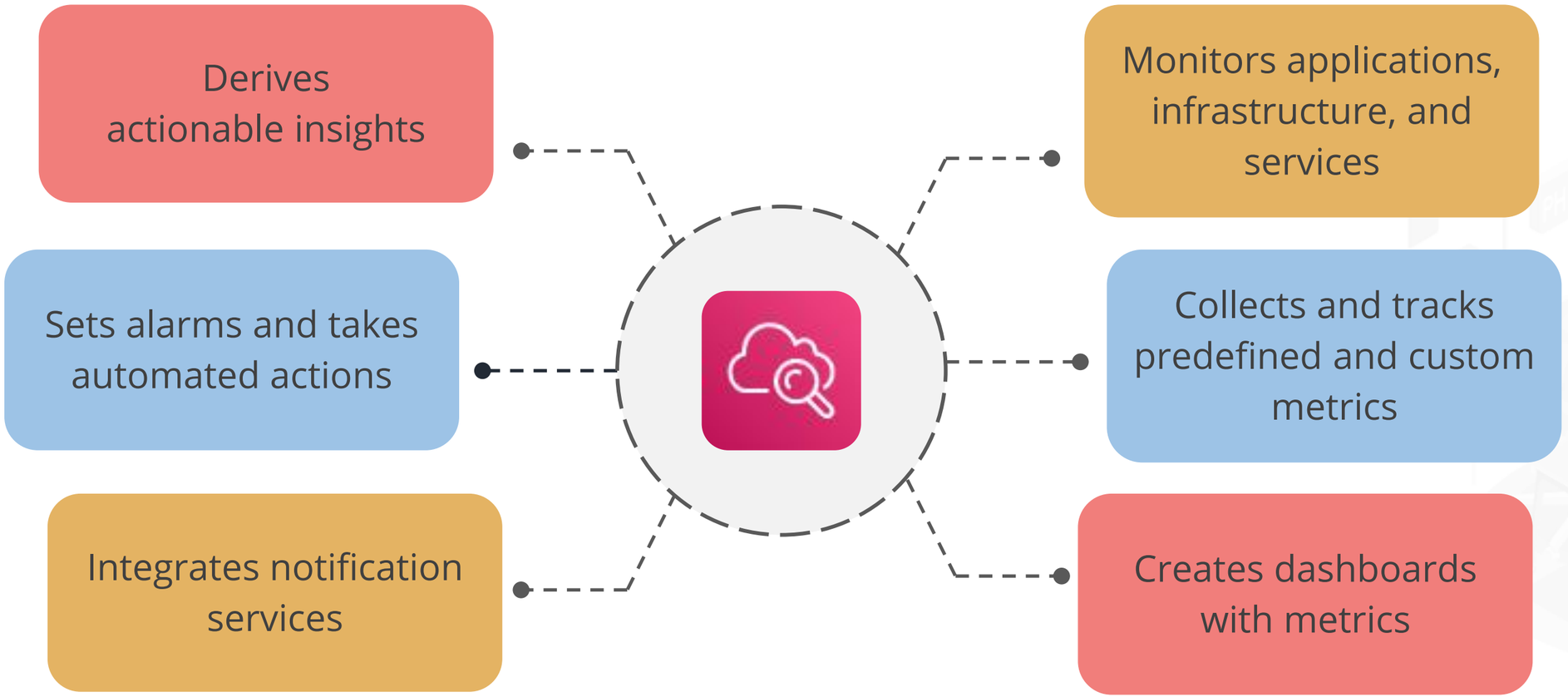
CloudWatch: Overview

1. CloudWatch is a monitoring and observational service used to monitor resources and applications that a user runs on AWS.

2. It helps users to collect and access all performances and operational data in the form of logs and metrics.

3. It provides system-wise visibility into resource utilization, application performance, and operational health.

Features of CloudWatch



CloudWatch Workflow

Control: It collects metrics, and logs from AWS services that run on AWS and on-premises servers.

1

Monitor: It provides a uniform operational view with dashboards.

2

Act: CloudWatch automates response to operational changes with events and autoscaling.

3

Analyze: It provides real-time analysis with CloudWatch Metric Math.

4

Compliance and Security: It is integrated with IAM to track users' activities on resources.

5

CloudWatch Metrics

AWS services provide metrics about performance.

- Basic (free)
- Detailed (paid)

Custom metrics are also supported.

- Metric data is available for 15 months
- Alarms are set against values for metrics

Metrics can be used for searching and graphing on dashboards.

CloudWatch Metrics

A time stamp is required for each metric data point.

The time stamp can be up to two weeks in the past and two hours in the future. In the absence of a time stamp from the user, CloudWatch generates one, based on the time the data point was received.

TIMESTAMP

CloudWatch Metrics

- DateTime objects with the full date, hours, minutes and seconds make up time stamps. It is advised to utilize Coordinated Universal Time (UTC).
- Based on the current time in UTC, CloudWatch alarms check metrics.
- Custom metrics sent to CloudWatch with time stamps different from the current UTC time can cause alarms to display the **Insufficient Data** state or result in delayed alarms.

CloudWatch Metrics

CloudWatch retains metric data as below:

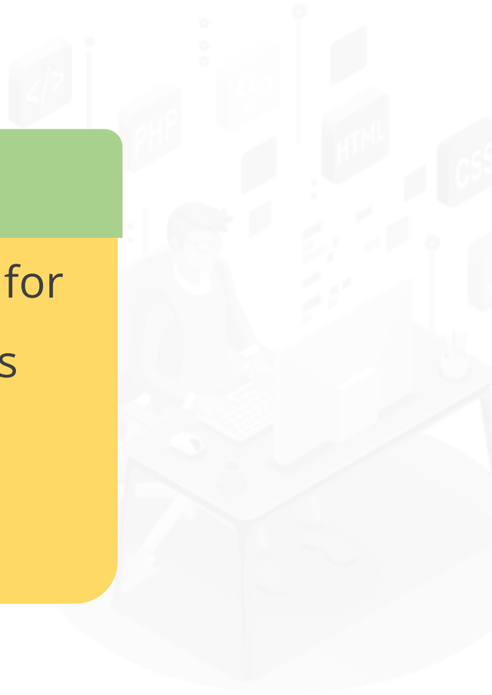
1. Data points with a period of fewer than 60 seconds are available for 3 hours.
2. Data points with a period of 60 seconds have an availability of 15 days.
3. Data points with a period of 300 seconds have an availability of 63 days.
4. Data points with a period of 3600 seconds are available for 455 days (15 months).

CloudWatch Metrics

Data points that are initially published within a shorter period are aggregated together for long-term storage.

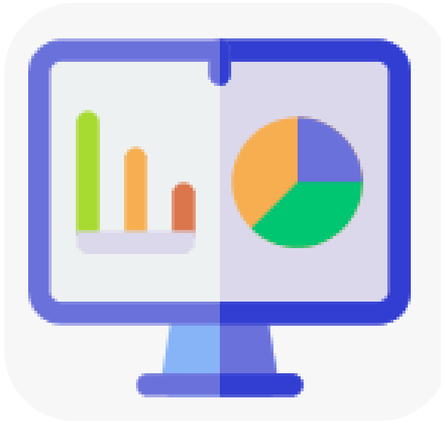
Example:

If the user collects data using a period of one minute, the data remains available for 15 days with a 1-minute resolution. After 15 days, this data is aggregated and is retrievable with a resolution of five minutes. After 63 days, the data is further aggregated and is available with a resolution of one hour.



CloudWatch Alarms

An alarm is used by the user to initiate actions automatically.



- An alarm monitors a single metric over a specified period and takes actions based on the metric's value relative to a threshold over time.
- An alert is issued to an Auto Scaling policy or an Amazon SNS subject as the action.

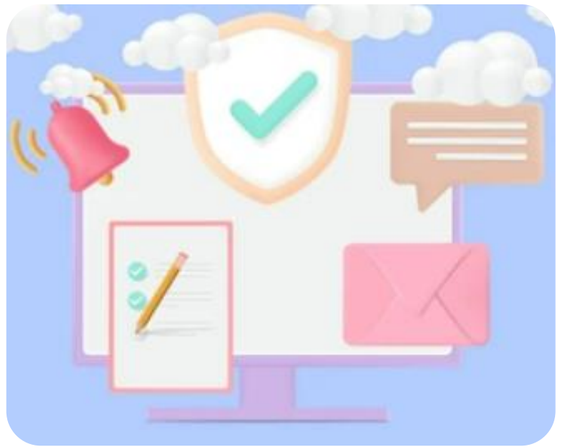
CloudWatch Alarms

The alarm invokes action for sustained state changes only.

CloudWatch alarms do not invoke actions when they are in a particular state. There must be a state change that is being maintained for a specified number of periods.

CloudWatch Alarms

The user should choose an alarm monitoring period that is larger than or equal to the metric's resolution when setting up an alarm.



There is a higher charge for high-resolution alarms.

CloudWatch Alarms

Example:

- Every five minutes, metrics for the user's instances are sent by Amazon EC2's basic monitoring.
- User needs to choose a duration when setting an alarm on a fundamental monitoring measure.
- If the user sets an alert for a high-resolution metric, they can choose between a standard alarm with a period of any multiple of 60 seconds.

CloudWatch Logs

The user can use Amazon CloudWatch Logs to monitor, store, and access log files from Amazon Elastic Compute Cloud (Amazon EC2) instances, AWS CloudTrail, Route 53, and other sources.

CloudWatch Logs

With CloudWatch Logs, users can centralize the logs from all systems, applications, and AWS services that they use in a single, highly scalable service.

Users can then easily view them, search them for specific error codes, filter them based on fields, or archive them securely for future analysis.

CloudWatch Logs

With CloudWatch Logs, users can view all their logs as a single, unified flow of events that are ordered by time, regardless of where they came from.

Users can query and sort their logs based on other dimensions, group them according to particular fields, build custom computations using a powerful query language, and view log data in dashboards.

Features of CloudWatch Logs

Firstly, CloudWatch logs provide information regarding the query of Log data.

- Users can use CloudWatch Logs Insight to search and analyze the log data.
- They can perform queries to effectively respond to operational issues.
- CloudWatch Logs Insights includes a purpose-built query language with powerful commands.

Features of CloudWatch Logs

Secondly, they help to monitor logs from Amazon EC2 instances.

Example:

- CloudWatch Logs can track the number of errors occurring in application logs and send a notification whenever the rate of errors exceeds the threshold specified by the user.

Features of CloudWatch Logs

Thirdly, they help to monitor AWS CloudTrail logged events.

The users can create alarms in CloudWatch and receive notifications of API activity as captured by CloudTrail and use the notification to perform troubleshooting.

Features of CloudWatch Logs

Next, CloudWatch logs help in log retention. Logs are retained forever by default, and they never expire.

Users can change the indefinite retention policy for each log group or select a retention duration between one day and 10 years.

Features of CloudWatch Logs

They archive log data so that users can store log data in long-lasting storage by using CloudWatch Logs.

It is easy to send rotated and non-rotated log data off of a host and into the log service using the CloudWatch Logs agent.

Features of CloudWatch Logs

Lastly, they also help to log Route 53 DNS queries. Users can record information about the DNS requests that Route 53 gets using CloudWatch Logs.

CloudWatch Alarms

Duration: 8 mins

Problem Statement:

You have been asked to launch an EC2 instance and stop it using CloudWatch Alarms.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Step to be followed:

1. Create the CloudWatch Alarm to stop the instance if CPU utilization is less than 30 percent

Evaluate CloudWatch Log Reports

Duration: 8 mins

Problem Statement:

You have been assigned a task to evaluate CloudWatch log reports and look for patterns in the data using metric filters and plotting the results in a graph.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Log in to your AWS lab
2. Open CloudWatch
3. In the Navigation pane, choose log groups
4. Select the log group
5. Select metric filter
6. Create metric
7. Graph the metric

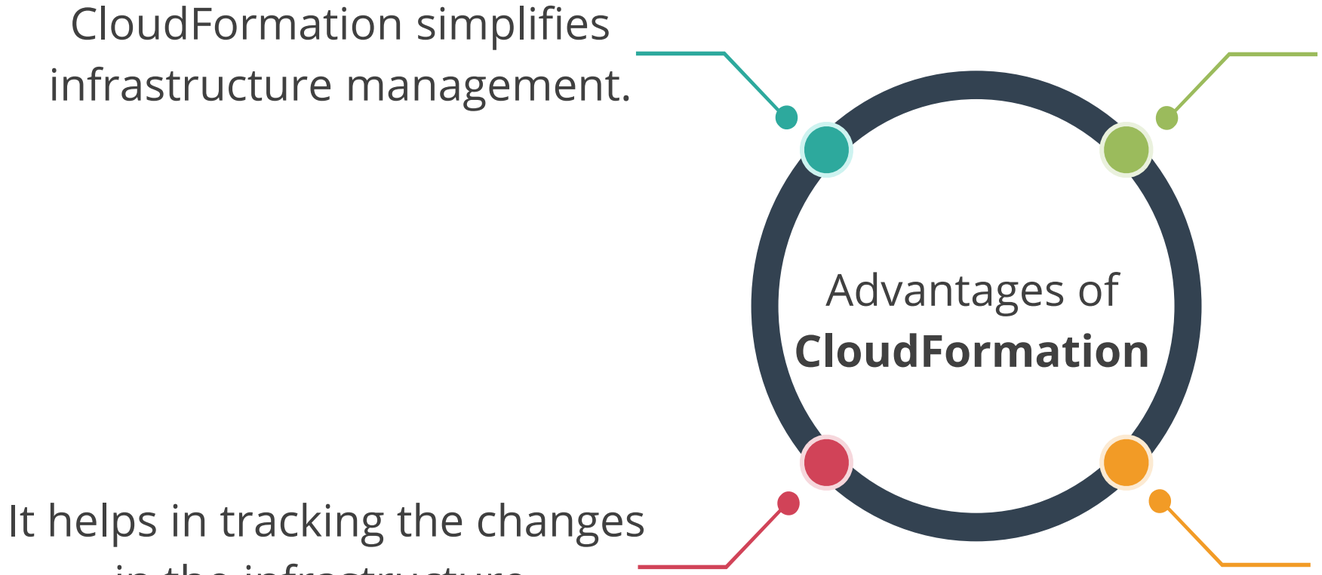
CloudFormation

Introduction to CloudFormation

AWS CloudFormation is a service that helps an end-user automate AWS services by creating templates.

The CloudFormation interprets the template and makes the API calls to create the resources.

Introduction to CloudFormation



Introduction to CloudFormation

Templates and **Stacks** are important elements of CloudFormation.

A **template** is a JSON or YAML formatted text file.

A **stack** is a collection of AWS resources that a user can manage as a single unit.

Types of Templates

Sample templates in JSON and YAML file format

Template in JSON format

```
{
  "AWSTemplateFormatVersion" : "version date",

"Description" : "JSON 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  }
}
```

Template in YAML format

```
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
```

Template Structure Overview

Sections of template are as follows:

Data tables (optional):
This section includes static configuration values, such as AMI names.

Outputs (optional):
This section includes the values that are returned whenever you view your stack's properties.

Parameters (optional):
This section includes input values that are provided while creating stacks.

Resources (required):
This section includes names and configurations of services to be added.

Template Structure Overview

Sections of template are as follows:

05

Conditions (optional): :

This section includes actions to be taken based on conditional statements, such as equals.

06

Mappings (optional): :

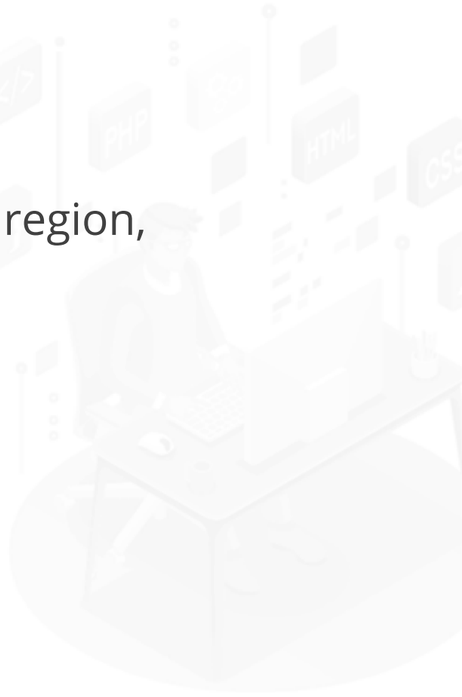
This section has set values based on a region, such as AMIs

.

07

Transform (optional): :

This section includes snippets from outside the main template.



Template Structure Overview

Template in JSON format

```
{"AWSTemplateFormatVersion" : "version date",

"Description" : "JSON 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  }}
```

The template contains the following section:

- Template Version
- Description
- Metadata
- Parameter
- Mapping
- Conditions
- Output
- Resources

Parameter Section Example

This example shows how to specify Input Parameters for a template:

```
"Parameters" : {

 "InstanceTypeParameter":  {

"Type" :  "String",

 "Default"  : "t2.micro",

"AllowedValues" : ["t2.micro", "m1.small","ml.large"],

"Description" : "Enter t2.micro, m1.small, or m1.large. Default is
t2.micro."
  }
}
```

Mapping Section Example

This example shows how to match a key to a corresponding set of named values:

```
"Mappings": {
    "Mapping01" : {
       "Key01" : {
         "Name":"Value01"
        }

      "Key02": {
         "Name" : "Value02"
         }
      "Key03": {
           "Name" : "Value03"
         }
     }
}
```

Condition Section

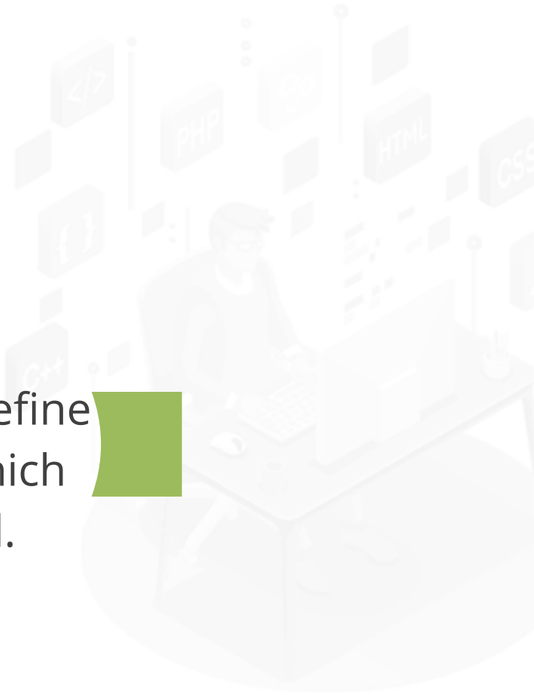
Properties of the condition section of the template are:

01

The condition statement is optional.

02

It contains statements that define the circumstances under which the resources are created.



Resources Section Example

This example shows how to declare the AWS resources that need to be included in the stack.

```
"Resources": {
        "SQSQ44CP0": {
            "Type": "AWS::SQS::Queue",
            "Properties": {},
            "Metadata": {
                "AWS::CloudFormation::Designer": {
                    "id": "3f3e280c-2e3b-43d2-bdaa-9c9a529f6de8"
                }
            }
        },
        "S3B4XB7F": {
            "Type": "AWS::S3::Bucket",
            "Properties": {},
            "Metadata": {
                "AWS::CloudFormation::Designer": {
                    "id": "e5cbfee9-294b-4688-afd2-43274d15a8bc"
```

Create an S3 Bucket Using CloudFormation

Duration: 8 mins

Problem Statement:

You have been asked to create an S3 bucket using CloudFormation

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Create an S3 bucket template
2. Create an S3 bucket stack using CloudFormation

Delete an Existing Stack

Duration: 8 mins

Problem Statement:

You have been to delete stack an existing stack

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Locate an existing stack and delete it

Update an Existing Stack

Duration: 8 mins

Problem Statement:

You have been asked to update an existing stack by deleting an S3 bucket.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Create an S3 bucket stack using CloudFormation
2. Delete an S3 bucket from the stack

Update Input Parameters in an Existing Stack

Duration: 8 mins

Problem Statement:

You have been asked to update the parameters for your existing template.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Log in to your AWS lab
2. Go to CloudFormation
3. Deploy a stack using a template
4. Update the template for a set of parameters
5. Test the updates

Update Deletion Policy to Protect User's Resources

Duration: 8 mins

Problem Statement:

You have been asked to protect a provisioned resource in the stack from deletions when the stack is deleted by updating the stack's deletion policy.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Log in to your AWS lab
2. Go to CloudFormation
3. Deploy a stack using a template
4. On the template, edit the Deletion policy to retain the specific resources
5. Delete the stack and test if the specified resources have been retained

TECHNOLOGY

Intrinsic Functions and Pseudo Parameters

simplilearn

Intrinsic Functions

Intrinsic functions are built-in functions that enable the users to assign values to properties that are available at runtime.

Examples of intrinsic function

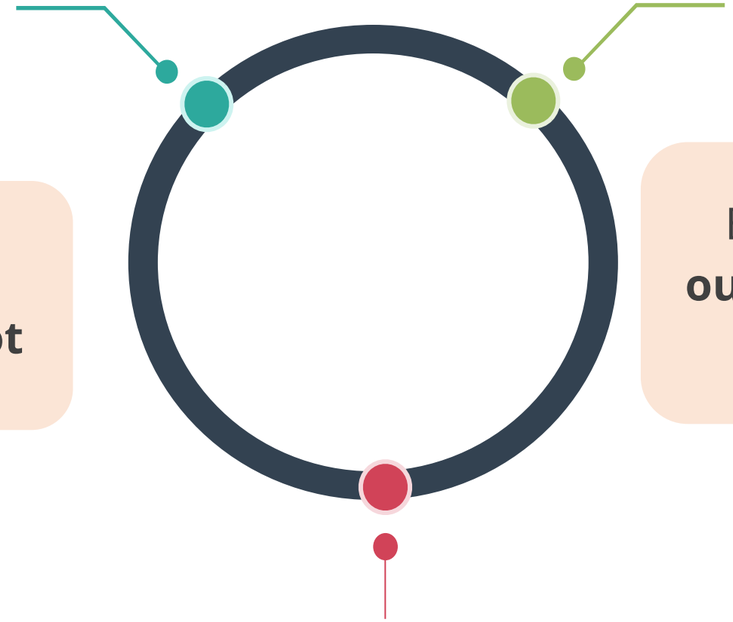
- Fn::Base64
- Fn::Cidr
- Fn::Join
- Fn::Select
- Fn::Split
- Fn::Transform
- Ref

Intrinsic Functions

The applications of intrinsic function are as follows:

Intrinsic functions can be used to create stack resources conditionally.

Examples:
Fn::If, Fn::Equals, and **Fn::Not**



They are used only in specific sections of the template.

Examples: **resource properties**, **outputs**, **metadata attributes**, and **update policy attributes**

They are used in templates to assign values to properties that are not available until runtime.

Intrinsic Functions

This example shows how to declare resources using **Ref intrinsic function:**

```
"My EIP" : {
    "Type" : "AWS::EC2::EIP",
    "Properties" : {
         "InstanceId" : {"Ref" : "MyEC2Instance"}
     }
  }
```

Pseudo Parameters

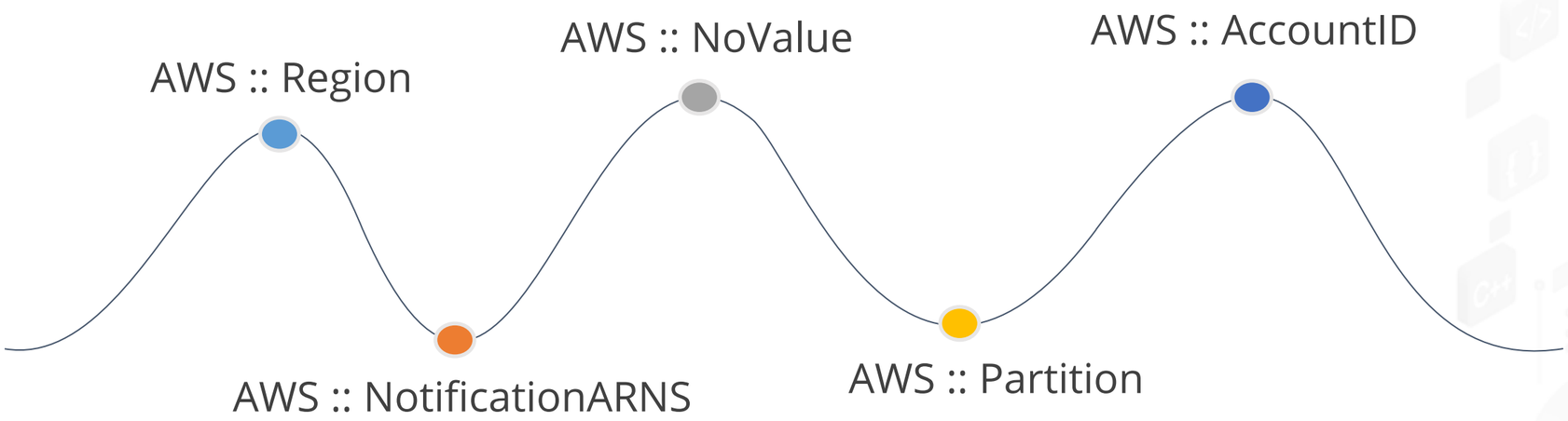
Pseudo parameters are those parameters that are predefined by AWS CloudFormation.

01 Pseudo parameters are not declared in the template.

02 Pseudo parameters can be referenced by using **Ref** intrinsic function.

Pseudo Parameters

The list of pseudo parameters are as follows:



Pseudo Parameters

This example shows how to assign the value of **AWS::Region** to the output value:

```
"Outputs" : {
    "MyStacksRegion" : { "Values" { "Ref" : "AWS::Region",} }
}
```

Create an Auto Scaling Group Using a Launch Template

Duration: 8 mins

Problem Statement:

You have been asked to create an Auto scaling group using a launch template

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Create a Launch template
2. Create an Auto Scaling Group

Create a Resource Group for Grouping Related Resources

Duration: 8 mins

Problem Statement:

You have been asked to implement AWS Resource Groups and Tag Editors.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed:

1. Launch multiple EC2 instances and create three tags on each EC2 instance
2. Create a resource group consisting of related resources

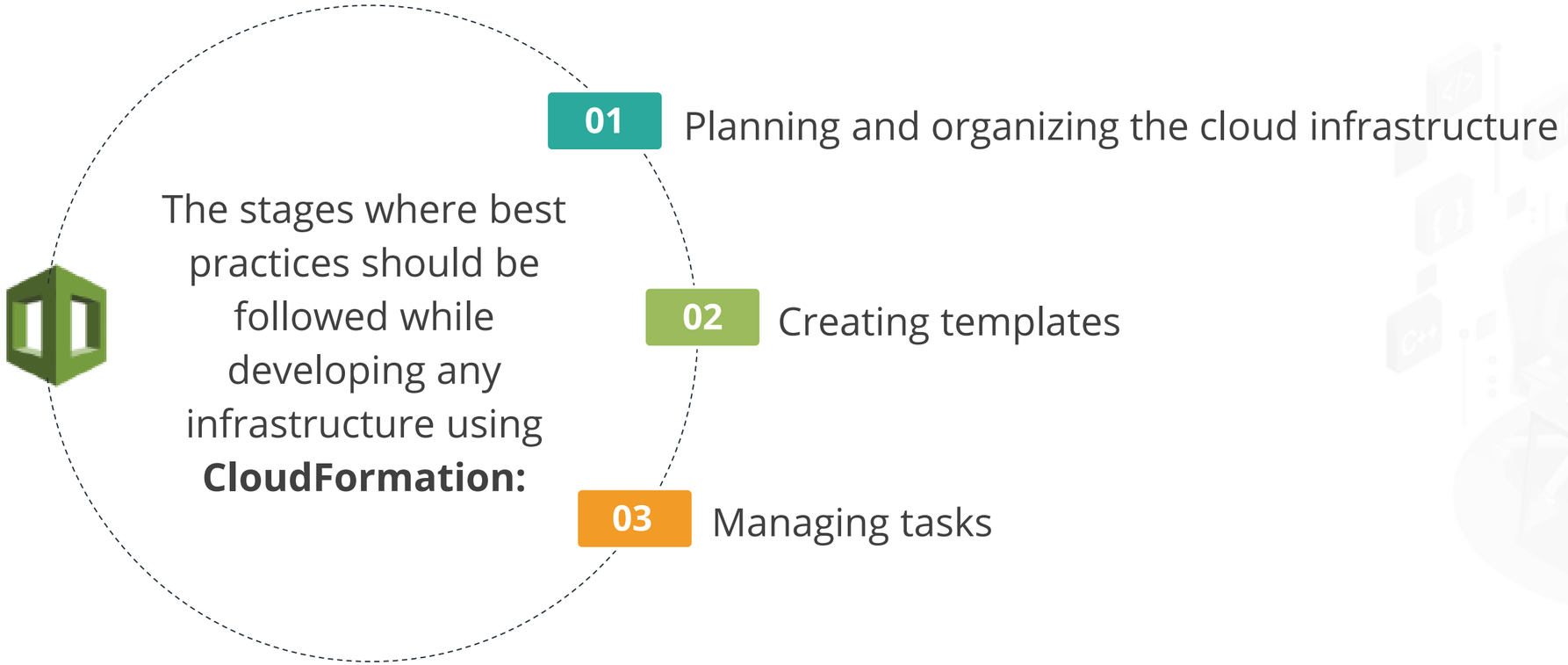
TECHNOLOGY

CloudFormation Best Practices

simplilearn

AWS CloudFormation Best Practices

Best practices are recommendations that can help in the effective utilization of **AWS CloudFormation** throughout its entire workflow.



AWS CloudFormation Best Practices

The list of practices to be followed while **planning and organization** the cloud infrastructure are as follows:

- Organize the stacks by life cycle and ownership
- Use cross-stack references to export shared resources
- Use IAM to control access
- Reuse templates to replicate stacks in multiple environments
- Verify quotas for all resource types
- Use modules to reuse resource configurations

AWS CloudFormation Best Practices

The list of practices to be followed while **creating templates** are as follows:

- Do not use credentials in a template
- Use AWS-specific parameter types
- Use parameter constraints
- Use **AWS::CloudFormation::Init** to deploy software applications on Amazon EC2 instances
- Use the latest helper scripts
- Use templates after validating them

AWS CloudFormation Best Practices

The list of practices to be followed while **managing tasks** are as follows:

- Manage all stack resources through AWS CloudFormation
- Create change sets before updating the stacks
- Use stack policies
- Use AWS CloudTrail to log AWS CloudFormation calls
- Use code reviews and revision controls to manage the templates
- Update the Amazon EC2 instances regularly

Key Takeaways

- CloudWatch is a monitoring and observational service used to monitor resources and applications that a user runs on AWS.
- AWS CloudFormation is a service that helps to model and set up the AWS resources in an efficient manner.
- Templates and stacks are the major elements of CloudFormation.
- Intrinsic functions are used to assign values to properties that are only available at runtime.

Create an Alarm Using CloudWatch

Duration: 30 Mins

Project agenda: To create an alarm using CloudWatch that will allow you to watch CloudWatch metrics (CPU Utilization) with a given threshold and receive notifications when the metrics fall outside the threshold levels that you configure.

Description: Launch 3 virtual machine instances (Linux). Perform tasks on these VMS of your choice. Set up a dashboard with metrics showing CPU Utilization of all 3 VMS.

Perform the following:

1. Launch 3 Linux VMs.
2. Connect SSH into VMs.
3. Perform Linux-related tasks on VM.
4. Work with CloudWatch services.
5. Select metrics for CPU utilization for all 3 VMs.
6. Create an alarm and send a notification through SNS.

LESSON-END PROJECT

TECHNOLOGY

Thank You

simplilearn