**What is AWS CloudFormation?**

AWS CloudFormation is a service that helps you model and set up your Amazon Web Services resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS. You create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and AWS CloudFormation takes care of provisioning and configuring those resources for you. You don't need to individually create and configure AWS resources and figure out what's dependent on what; AWS CloudFormation handles all of that. The following scenarios demonstrate how AWS CloudFormation can help.

## Simplify Infrastructure Management

For a scalable web application that also includes a back-end database, you might use an Auto Scaling group, an Elastic Load Balancing load balancer, and an Amazon Relational Database Service database instance. Normally, you might use each individual service to provision these resources. And after you create the resources, you would have to configure them to work together. All these tasks can add complexity and time before you even get your application up and running.

Instead, you can create or modify an existing AWS CloudFormation template. A template describes all of your resources and their properties. When you use that template to create an AWS CloudFormation stack, AWS CloudFormation provisions the Auto Scaling group, load balancer, and database for you. After the stack has been successfully created, your AWS resources are up and running. You can delete the stack just as easily, which deletes all the resources in the stack. By using AWS CloudFormation, you easily manage a collection of resources as a single unit.

## Quickly Replicate Your Infrastructure

If your application requires additional availability, you might replicate it in multiple regions so that if one region becomes unavailable, your users can still use your application in other regions. The challenge in replicating your application is that it also requires you to replicate your resources. Not only do you need to record all the resources that your application requires, but you must also provision and configure those resources in each region.

When you use AWS CloudFormation, you can reuse your template to set up your resources consistently and repeatedly. Just describe your resources once and then provision the same resources over and over in multiple regions.

## Easily Control and Track Changes to Your Infrastructure

In some cases, you might have underlying resources that you want to upgrade incrementally. For example, you might change to a higher performing instance type in your Auto Scaling launch configuration so that you can reduce the maximum number of instances in your Auto Scaling group. If problems occur after you complete the update, you might need to roll back your infrastructure to the original settings. To do this manually, you not only have to remember which resources were changed, you also have to know what the original settings were.

When you provision your infrastructure with AWS CloudFormation, the AWS CloudFormation template describes exactly what resources are provisioned and their settings. Because these templates are text files, you simply track differences in your templates to track changes to your infrastructure, similar to the way developers control revisions to source code. For example, you can use a version control system with your templates so that you know exactly what changes were made, who made them, and when. If at any point you need to reverse changes to your infrastructure, you can use a previous version of your template.

# AWS CloudFormation Concepts

When you use AWS CloudFormation, you work with templates and stacks. You create templates to describe your AWS resources and their properties. Whenever you create a stack, AWS CloudFormation provisions the resources that are described in your template.

**Topics**

* [Templates](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-whatis-concepts.html#w2ab2b5c15b7)
* [Stacks](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-whatis-concepts.html#w2ab2b5c15b9)
* [Change Sets](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-whatis-concepts.html#w2ab2b5c15c11)

## Templates

An AWS CloudFormation template is a JSON or YAML formatted text file. You can save these files with any extension, such as .json, .yaml, .template, or .txt. AWS CloudFormation uses these templates as blueprints for building your AWS resources. For example, in a template, you can describe an Amazon EC2 instance, such as the instance type, the AMI ID, block device mappings, and its Amazon EC2 key pair name. Whenever you create a stack, you also specify a template that AWS CloudFormation uses to create whatever you described in the template.

For example, if you created a stack with the following template, AWS CloudFormation provisions an instance with an ami-2f726546 AMI ID, t1.micro instance type, testkey key pair name, and an Amazon EBS volume.

**Example JSON**

{

"AWSTemplateFormatVersion" : "2010-09-09",

"Description" : "A sample template",

"Resources" : {

"MyEC2Instance" : {

"Type" : "AWS::EC2::Instance",

"Properties" : {

"ImageId" : "ami-2f726546",

"InstanceType" : "t1.micro",

"KeyName" : "testkey",

"BlockDeviceMappings" : [

{

"DeviceName" : "/dev/sdm",

"Ebs" : {

"VolumeType" : "io1",

"Iops" : "200",

"DeleteOnTermination" : "false",

"VolumeSize" : "20"

}

}

]

}

}

}

}

**Example YAML**

AWSTemplateFormatVersion: "2010-09-09"

Description: A sample template

Resources:

MyEC2Instance:

Type: "AWS::EC2::Instance"

Properties:

ImageId: "ami-2f726546"

InstanceType: t1.micro

KeyName: testkey

BlockDeviceMappings:

-

DeviceName: /dev/sdm

Ebs:

VolumeType: io1

Iops: 200

DeleteOnTermination: false

VolumeSize: 20

You can also specify multiple resources in a single template and configure these resources to work together. For example, you can modify the previous template to include an Elastic IP (EIP) and associate it with the Amazon EC2 instance, as shown in the following example:

**Example JSON**

{

"AWSTemplateFormatVersion" : "2010-09-09",

"Description" : "A sample template",

"Resources" : {

"MyEC2Instance" : {

"Type" : "AWS::EC2::Instance",

"Properties" : {

"ImageId" : "ami-2f726546",

"InstanceType" : "t1.micro",

"KeyName" : "testkey",

"BlockDeviceMappings" : [

{

"DeviceName" : "/dev/sdm",

"Ebs" : {

"VolumeType" : "io1",

"Iops" : "200",

"DeleteOnTermination" : "false",

"VolumeSize" : "20"

}

}

]

}

},

"MyEIP" : {

"Type" : "AWS::EC2::EIP",

"Properties" : {

"InstanceId" : {"Ref": "MyEC2Instance"}

}

}

}

}

**Example YAML**

AWSTemplateFormatVersion: "2010-09-09"

Description: A sample template

Resources:

MyEC2Instance:

Type: "AWS::EC2::Instance"

Properties:

ImageId: "ami-2f726546"

InstanceType: t1.micro

KeyName: testkey

BlockDeviceMappings:

-

DeviceName: /dev/sdm

Ebs:

VolumeType: io1

Iops: 200

DeleteOnTermination: false

VolumeSize: 20

MyEIP:

Type: AWS::EC2::EIP

Properties:

InstanceId: !Ref MyEC2Instance

The previous templates are centered around a single Amazon EC2 instance; however, AWS CloudFormation templates have additional capabilities that you can use to build complex sets of resources and reuse those templates in multiple contexts. For example, you can add input parameters whose values are specified when you create an AWS CloudFormation stack. In other words, you can specify a value like the instance type when you create a stack instead of when you create the template, making the template easier to reuse in different situations.

## Stacks

When you use AWS CloudFormation, you manage related resources as a single unit called a stack. You create, update, and delete a collection of resources by creating, updating, and deleting stacks. All the resources in a stack are defined by the stack's AWS CloudFormation template. Suppose you created a template that includes an Auto Scaling group, Elastic Load Balancing load balancer, and an Amazon Relational Database Service (Amazon RDS) database instance. To create those resources, you create a stack by submitting the template that you created, and AWS CloudFormation provisions all those resources for you. You can work with stacks by using the AWS CloudFormation [console](https://console.aws.amazon.com/cloudformation/), [API](http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/), or [AWS CLI](http://docs.aws.amazon.com/cli/latest/reference/cloudformation).

## Change Sets

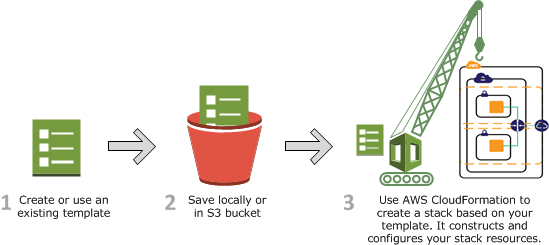
If you need to make changes to the running resources in a stack, you update the stack. Before making changes to your resources, you can generate a change set, which is summary of your proposed changes. Change sets allow you to see how your changes might impact your running resources, especially for critical resources, before implementing them.

For example, if you change the name of an Amazon RDS database instance, AWS CloudFormation will create a new database and delete the old one. You will lose the data in the old database unless you've already backed it up. If you generate a change set, you will see that your change will cause your database to be replaced, and you will be able to plan accordingly before you update your stack. For more information, see [Updating Stacks Using Change Sets](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html).

# How Does AWS CloudFormation Work?

When you create a stack, AWS CloudFormation makes underlying service calls to AWS to provision and configure your resources. Note that AWS CloudFormation can perform only actions that you have permission to do. For example, to create EC2 instances by using AWS CloudFormation, you need permissions to create instances. You'll need similar permissions to terminate instances when you delete stacks with instances. You use [AWS Identity and Access Management](http://docs.aws.amazon.com/IAM/latest/UserGuide/) (IAM) to manage permissions.

The calls that AWS CloudFormation makes are all declared by your template. For example, suppose you have a template that describes an EC2 instance with a t1.micro instance type. When you use that template to create a stack, AWS CloudFormation calls the Amazon EC2 create instance API and specifies the instance type as t1.micro. The following diagram summarizes the AWS CloudFormation workflow for creating stacks.



1. You can design an AWS CloudFormation template (a JSON or YAML-formatted document) in [AWS CloudFormation Designer](https://console.aws.amazon.com/cloudformation/designer) or write one in a text editor. You can also choose to use a provided template. The template describes the resources you want and their settings. For example, suppose you want to create an EC2 instance. Your template can declare an EC2 instance and describe its properties, as shown in the following example:

**Example JSON Syntax**

{

"AWSTemplateFormatVersion" : "2010-09-09",

"Description" : "A simple EC2 instance",

"Resources" : {

"MyEC2Instance" : {

"Type" : "AWS::EC2::Instance",

"Properties" : {

"ImageId" : "ami-2f726546",

"InstanceType" : "t1.micro"

}

}

}

}

**Example YAML Syntax**

AWSTemplateFormatVersion: '2010-09-09'

Description: A simple EC2 instance

Resources:

MyEC2Instance:

Type: AWS::EC2::Instance

Properties:

ImageId: ami-2f726546

InstanceType: t1.micro

1. Save the template locally or in an S3 bucket. If you created a template, save it with any file extension like .json, .yaml, or .txt.
2. Create an AWS CloudFormation stack by specifying the location of your template file , such as a path on your local computer or an Amazon S3 URL. If the template contains parameters, you can specify input values when you create the stack. Parameters enable you to pass in values to your template so that you can customize your resources each time you create a stack.

You can create stacks by using the AWS CloudFormation [console](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-console-create-stack.html), [API](http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/API_CreateStack.html), or [AWS CLI](http://docs.aws.amazon.com/cli/latest/reference/cloudformation/create-stack.html).

Note

If you specify a template file stored locally, AWS CloudFormation uploads it to an S3 bucket in your AWS account. AWS CloudFormation creates a bucket for each region in which you upload a template file. The buckets are accessible to anyone with Amazon Simple Storage Service (Amazon S3) permissions in your AWS account. If a bucket created by AWS CloudFormation is already present, the template is added to that bucket.

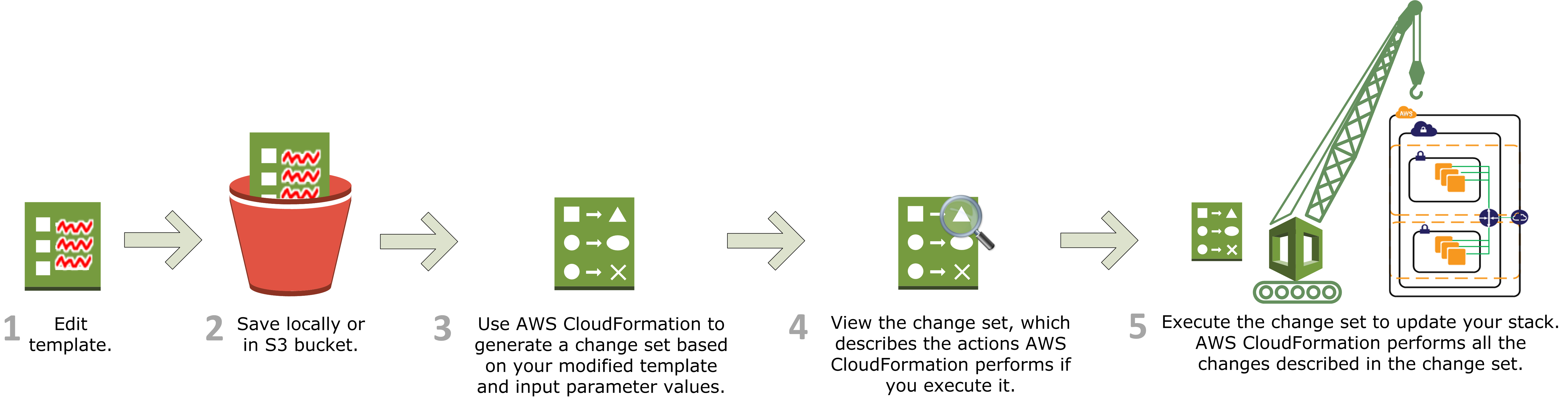
You can use your own bucket and manage its permissions by manually uploading templates to Amazon S3. Then whenever you create or update a stack, specify the Amazon S3 URL of a template file.

AWS CloudFormation provisions and configures resources by making calls to the AWS services that are described in your template.

After all the resources have been created, AWS CloudFormation reports that your stack has been created. You can then start using the resources in your stack. If stack creation fails, AWS CloudFormation rolls back your changes by deleting the resources that it created.

## Updating a Stack with Change Sets

When you need to update your stack's resources, you can modify the stack's template. You don't need to create a new stack and delete the old one. To update a stack, create a change set by submitting a modified version of the original stack template, different input parameter values, or both. AWS CloudFormation compares the modified template with the original template and generates a change set. The change set lists the proposed changes. After reviewing the changes, you can execute the change set to update your stack or you can create a new change set. The following diagram summarizes the workflow for updating a stack.



Important

Updates can cause interruptions. Depending on the resource and properties that you are updating, an update might interrupt or even replace an existing resource. For more information, see [AWS CloudFormation Stacks Updates](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks.html).

1. You can modify an AWS CloudFormation stack template by using [AWS CloudFormation Designer](https://console.aws.amazon.com/cloudformation/designer) or a text editor. For example, if you want to change the instance type for an EC2 instance, you would change the value of the InstanceType property in the original stack's template.

For more information, see [Modifying a Stack Template](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-get-template.html).

1. Save the AWS CloudFormation template locally or in an S3 bucket.
2. Create a change set by specifying the stack that you want to update and the location of the modified template, such as a path on your local computer or an Amazon S3 URL. If the template contains parameters, you can specify values when you create the change set.

For more information about creating change sets, see [Updating Stacks Using Change Sets](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html).

Note

If you specify a template that is stored on your local computer, AWS CloudFormation automatically uploads your template to an S3 bucket in your AWS account.

1. View the change set to check that AWS CloudFormation will perform the changes that you expect. For example, check whether AWS CloudFormation will replace any critical stack resources. You can create as many change sets as you need until you have included the changes that you want.

Important

Change sets don't indicate whether your stack update will be successful. For example, a change set doesn't check if you will surpass an account [limit](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cloudformation-limits.html), if you're updating a [resource](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html) that doesn't support updates, or if you have insufficient [permissions](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-iam-template.html) to modify a resource, all of which can cause a stack update to fail.

1. Execute the change set that you want to apply to your stack. AWS CloudFormation updates your stack by updating only the resources that you modified and signals that your stack has been successfully updated. If the stack updates fails, AWS CloudFormation rolls back changes to restore the stack to the last known working state.

## Deleting a Stack

When you delete a stack, you specify the stack to delete, and AWS CloudFormation deletes the stack and all the resources in that stack. You can delete stacks by using the AWS CloudFormation [console](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-console-delete-stack.html), [API](http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/API_DeleteStack.html), or [AWS CLI](http://docs.aws.amazon.com/cli/latest/reference/cloudformation/delete-stack.html).

If you want to delete a stack but want to retain some resources in that stack, you can use a [deletion policy](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-attribute-deletionpolicy.html) to retain those resources.

After all the resources have been deleted, AWS CloudFormation signals that your stack has been successfully deleted. If AWS CloudFormation cannot delete a resource, the stack will not be deleted. Any resources that haven't been deleted will remain until you can successfully delete the stack.

**Setting Up**

Before you start using AWS CloudFormation, you might need to know what IAM permissions you need, how to start logging AWS CloudFormation API calls, or what endpoints to use. The following topics provide this information so that you can start using AWS CloudFormation.

**Topics**

* [Signing Up for an AWS Account and Pricing](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-sign-up-for-aws.html)
* [Controlling Access with AWS Identity and Access Management](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-iam-template.html)
* [Logging AWS CloudFormation API Calls in AWS CloudTrail](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-api-logging-cloudtrail.html)
* [AWS CloudFormation Limits](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cloudformation-limits.html)
* [AWS CloudFormation Endpoints](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-endpoints.html)
* [AWS CloudFormation and VPC Endpoints](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-vpce-bucketnames.html)

# Signing Up for an AWS Account and Pricing

Before you can use AWS CloudFormation or any Amazon Web Services, you must first sign up for an AWS account.

**To sign up for an AWS account**

1. Open <https://aws.amazon.com/>, and then choose **Create an AWS Account**.

Note

This might be unavailable in your browser if you previously signed into the AWS Management Console. In that case, choose **Sign in to a different account**, and then choose **Create a new AWS account**.

1. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

After signing up for an AWS account, you can use AWS CloudFormation through the [AWS Management Console](https://console.aws.amazon.com/cloudformation/), [AWS CloudFormation API](http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/), or [AWS CLI](http://docs.aws.amazon.com/cli/latest/reference/cloudformation).

## Pricing

AWS CloudFormation is a free service; however, you are charged for the AWS resources you include in your stacks at the current rates for each. For more information about AWS pricing, go to the detail page for each product on [http://aws.amazon.com](http://aws.amazon.com/).