Cloud Control API

User Guide

Cloud Control API: User Guide

Table of Contents

What is AWS Cloud Control API?

Use AWS Cloud Control API to create, read, update, delete, and list (CRUD-L) your cloud resources that belong to a wide range of services—both AWS and third-party. With the Cloud Control API standardized set of application programming interfaces (APIs), you can perform CRUD-L operations on any supported resources in your AWS account. Using Cloud Control API, you won't have to generate code or scripts specific to each individual service responsible for those resources.

Topics

Are you a first-time Cloud Control API user?

If you're a first-time user of Cloud Control API, we recommend that you begin by reading the following sections:

- Setting up
- Getting started

Features of Cloud Control API

Cloud Control API provides you with consistent control over the resources in your AWS account by offering a standardized way of accessing and provisioning those resources. It provides a uniform programmatic interface for making calls directly to the various resource types available in your AWS account, without you having to familiarize yourself with APIs of the underlying web services.

Related services

Similar to Cloud Control API, AWS CloudFormation also uses resource types to call underlying web services APIs to provision those resources when you place such a request in your account. However, CloudFormation focuses on providing resource management, by treating infrastructure as code. Using CloudFormation, you can author declarative templates that include multiple resources and their dependencies, and then provision those resources as a *stack*. A stack is a single unit that you then manage through AWS CloudFormation. You can also centrally manage and provision stacks across multiple AWS accounts and AWS Regions. To be managed through CloudFormation, a resource must be created as part of a stack or imported into a stack. For more information, see the *AWS CloudFormation User Guide*.

Accessing Cloud Control API

Cloud Control API provides API operations for generating create, read, update, delete, and list (CRUD-L) resource requests in addition to tracking and managing those requests. You use the AWS Command Line Interface (AWS CLI) for Cloud Control API operations.

The following table shows the Cloud Control API operations you can use to generate CRUD-L resource requests.

API operation	AWS CLI command
CreateResource	create-resource
DeleteResource	delete-resource
GetResource	get-resource
ListResources	list-resources
UpdateResource	update-resource

The following table shows the Cloud Control API operations that you can use to track and manage resource requests while they're in process.

API operation	AWS CLI command
CancelResourceRequest	cancel-resource-request
GetResourceRequestStatus	get-resource-request-status
ListResourceRequests	list-resource-requests

How Cloud Control API works

Cloud Control API provides you with centralized control over the resources in your AWS account and a consistent way of accessing and provisioning those resources. It provides a uniform programmatic interface for making calls directly to the various resource types available in your AWS account.

A *resource type* represents an artifact that can be provisioned through a web service: an Amazon Elastic Compute Cloud (Amazon EC2) instance, an Amazon Relational Database Service (Amazon RDS) database instance, an AWS Identity and Access Management (IAM) policy, or even an entire web application. Each resource type uses a standardized syntax to support some or all of the following lifecycle events: create, read, update, delete, and list (CRUD-L). You can directly invoke these CRUD-L event handlers using Cloud Control API as a consistent set of APIs.

Amazon has published several hundred resource types representing offerings across AWS web services. Now, third-party publishers can make their own resource types available for use as well. Any resource type developed using the AWS CloudFormation CLI open-source tool is automatically supported by Cloud Control API.

Each resource type is defined by its *resource type schema*. This document is compliant with the JSON Schema open standard, and includes:

- A complete list of each resource property and its associated metadata, including whether the property is required, data type, and value constraints.
- The CRUD-L events that the resource type supports, and the permissions necessary for Cloud Control API to invoke each supported event handler.

When you create or update a resource, you specify JSON that represents the properties and property values you want to set for the resource. Cloud Control API handles the actual calls to the underlying web services to perform the requested changes. For read requests, Cloud Control API returns JSON that represents the current state of the specified resource. For list requests, Cloud Control API returns either the resource identifier or JSON that represents the current state of the specified resources.

You can use Cloud Control API to perform operations on existing resources, whether or not those resources were created using Cloud Control API. For example, you could use Cloud Control API to return property information about each AWS Lambda function in your AWS account.

For a brief tutorial on how to use Cloud Control API to perform resource operations, see Getting started.

For more information about resource types and how to use them with Cloud Control API, see Using resource types.

Setting up AWS Cloud Control API

To use AWS Cloud Control API, you'll need to have an AWS account in which you have set yourself up as an AWS Identity and Access Management (IAM) administrator user.

Topics

Sign up for AWS

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Getting started with Cloud Control API

Use this short tutorial to get started performing resource operations with AWS Cloud Control API. You'll learn the basics of using Cloud Control API to create, read, update, delete, and list resources.

Topics

Step 1: Create a resource

For this tutorial, create a resource of type `AWS::Logs::LogGroup`. Name this log group **`CloudControlExample`**, and set the retention policy on it to 90 days.

1. In the AWS Command Line Interface (AWS CLI), run the `create-resource` command with the following parameters:

   - Specify the `type-name` as `AWS::Logs::LogGroup`.
   - Specify the `desired-state` as a string containing JSON that sets the desired properties:

     `{\"LogGroupName\": \"CloudControlExample\",\"RetentionInDays\":90}`

   ```
   aws cloudcontrol create-resource --type-name AWS::Logs::LogGroup --desired-state
    "{\"LogGroupName\": \"CloudControlExample\",\"RetentionInDays\":90}"
   ```

   Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request.

   ```
   {
       "ProgressEvent": {
           "EventTime": "2021-08-26T22:07:23.347Z",
           "TypeName": "AWS::Logs::LogGroup",
           "OperationStatus": "IN_PROGRESS",
           "Operation": "CREATE",
           "Identifier": "CloudControlExample",
           "RequestToken": "758f4a4e-fef4-491a-9b07-00123456789"
       }
   }
   ```

2. To track the status of your resource operation request, run the `get-resource-request-status` command with the following parameter:

   - Specify the `request-token` parameter as the `RequestToken` property value returned in the `ProgressEvent` object.

```
aws cloudcontrol get-resource-request-status --request-token 758f4a4e-
fef4-491a-9b07-00123456789
```

Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request. When Cloud Control API has successfully created the resource, it sets the `OperationStatus` value to `SUCCESS`.

```
{
    "ProgressEvent": {
        "EventTime": "2021-08-26T22:29:23.326Z",
        "TypeName": "AWS::Logs::LogGroup",
        "OperationStatus": "SUCCESS",
        "Operation": "CREATE",
        "Identifier": "CloudControlExample",
        "RequestToken": "758f4a4e-fef4-491a-9b07-00123456789"
    }
}
```

Step 2: Read (describe) a resource

Next, read the current state of the resource you just created.

- In the AWS CLI, run the `get-resource` command with the following parameter:
  - Specify `identifier` as the `identifier` property value returned in the `ProgressEvent` object when you created the resource. In this case, it's `CloudControlExample`, the name you specified for the log group.

```
aws cloudcontrol get-resource --type-name AWS::Logs::LogGroup --identifier
 CloudControlExample
```

Cloud Control API returns detailed information about the resource's current state, including a model of its properties and settings. In this case, this includes a property, `Arn`, that was generated by Amazon CloudWatch Events when the resource was created.

```
{
    "TypeName": "AWS::Logs::LogGroup",
    "ResourceDescription": {
        "Identifier": "CloudControlExample",
        "ResourceModel": "{\"RetentionInDays\":90,\"LogGroupName\":
\"CloudControlExample\",\"Arn\":\"arn:aws:logs:us-west-2:090123456789:log-
group:CloudControlExample:*\"}"
    }
}
```

Step 3: Update a resource

Next, update your log group to double the retention policy to 180 days.

1. In the AWS CLI, run the `update-resource` command with the following parameter:

- Specify the `type-name` as `AWS::Logs::LogGroup`.
- Specify `identifier` as the `identifier` property value returned in the `ProgressEvent` object when you created the resource. In this case, it's `CloudControlExample`, the name you specified for the log group.
- Specify the `patch-document` parameter as a string containing JSON that represents a replacement operation that updates the retention policy to 180 days.

  ```
  [{\"op\":\"replace\",\"path\":\"/RetentionInDays\",\"value\":180}]
  ```

  For detailed information about composing patch documents, see Composing the patch document.

```
aws cloudcontrol update-resource --type-name AWS::Logs::LogGroup --identifier
 CloudControlExample --patch-document "[{\"op\":\"replace\",\"path\":\"/RetentionInDays
\",\"value\":180}]"
```

Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request.

```
{
    "ProgressEvent": {
        "EventTime": "2021-08-26T22:29:22.547Z",
        "ResourceModel": "{\"RetentionInDays\":180,\"LogGroupName\":
\"CloudControlExample\"}",
        "TypeName": "AWS::Logs::LogGroup",
        "OperationStatus": "IN_PROGRESS",
        "Operation": "UPDATE",
        "Identifier": "CloudControlExample",
        "RequestToken": "2026055d-f21c-4b50-bd40-111111111111"
    }
}
```

2. To track the status of your resource operation request, run the `get-resource-request-status` command with the following parameter:

   - Specify the `request-token` parameter as the `RequestToken` property value returned in the `ProgressEvent` object.

```
aws cloudcontrol get-resource-request-status --request-token 2026055d-f21c-4b50-
bd40-111111111111
```

Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request. When Cloud Control API has successfully updated the resource, it sets the `OperationStatus` value to `SUCCESS`.

```
{
    "ProgressEvent": {
        "EventTime": "2021-08-26T22:29:23.326Z",
        "TypeName": "AWS::Logs::LogGroup",
        "OperationStatus": "SUCCESS",
        "Operation": "UPDATE",
        "Identifier": "CloudControlExample",
        "RequestToken": "2026055d-f21c-4b50-bd40-111111111111"
    }
}
```

Step 4: List all resources of a certain type

Next, use Cloud Control API to discover resources in your AWS account.

- In the AWS CLI, run the `list-resources` command with the following parameter:
  - Specify the `type-name` as `AWS::Logs::LogGroup`.

  ```
  aws cloudcontrol list-resources --type-name AWS::Logs::LogGroup
  ```

  Cloud Control API returns a list of the `AWS::Logs::LogGroup` resources in your account, by primary identifier. This includes `CloudControlExample`, the resource you created as part of this tutorial, in addition to any other log groups that already exist in your account. Also, for `AWS::Logs::LogGroup` resources, the information returned by `list-resources` includes the properties for each resource.

  ```
  {
      "TypeName": "AWS::Logs::LogGroup",
      "ResourceDescriptions": [
          {
              "Identifier": "CloudControlExample",
              "Properties": "{\"RetentionInDays\":180,\"LogGroupName\":
  \"CloudControlExample\",\"Arn\":\"arn:aws:logs:us-west-2:090123456789:log-
  group:CloudControlExample:*\"}"
          },
          {
              "Identifier": "AnotherLogGroupResourceExample",
              "Properties": "{\"RetentionInDays\":90,\"LogGroupName\":
  \"AnotherLogGroupResourceExample\",\"Arn\":\"arn:aws:logs:us-west-2:011111111111:log-
  group:AnotherLogGroupResourceExample:*\"}"
          },
      ]
  }
  ```

Step 5: Delete a resource

Finally, delete your log group to clean up from this tutorial.

1. In the AWS CLI, run the `delete-resource` command with the following parameter:
   - Specify the `type-name` as `AWS::Logs::LogGroup`.
   - Specify `identifier` as the `identifier` property value returned in the `ProgressEvent` object when you created the resource. In this case, it's **`CloudControlExample`**, the name you specified for the log group.

   ```
   aws cloudcontrol delete-resource --type-name AWS::Logs::LogGroup --identifier
    CloudControlExample
   ```

   Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request.

   ```
   {
       "ProgressEvent": {
   ```

```
            "EventTime": "2021-08-26T22:50:20.037Z",
            "TypeName": "AWS::Logs::LogGroup",
            "OperationStatus": "IN_PROGRESS",
            "Operation": "DELETE",
            "Identifier": "CloudControlExample",
            "RequestToken": "bb0ed9cd-84f9-44c2-b638-000000000000"
        }
}
```

2. To track the status of your resource operation request, run the `get-resource-request-status` command with the following parameter:

    - Specify the `request-token` parameter as the `RequestToken` property value returned in the `ProgressEvent` object.

    ```
    aws cloudcontrol get-resource-request-status --request-token 2026055d-f21c-4b50-
    bd40-111111111111
    ```

    Cloud Control API returns a `ProgressEvent` object that contains information about the status of your resource operation request. When Cloud Control API has successfully deleted the resource, it sets the `OperationStatus` value to `SUCCESS`.

    ```
    {
        "ProgressEvent": {
            "EventTime": "2021-08-26T22:50:20.831Z",
            "TypeName": "AWS::Logs::LogGroup",
            "OperationStatus": "SUCCESS",
            "Operation": "DELETE",
            "Identifier": "CloudControlExample",
            "RequestToken": "bb0ed9cd-84f9-44c2-b638-000000000000"
        }
    }
    ```

Next steps

For detailed information and examples on using Cloud Control API with resources, see Performing resource operations.

Security in AWS Cloud Control API

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Cloud Control API, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations

AWS CloudFormation provides the security architecture for Cloud Control API; because of this, you will need to configure CloudFormation to meet your security and compliance objectives when using Cloud Control API. Refer to the Security section in the *AWS CloudFormation User Guide* to help you understand how to apply the shared responsibility model when using AWS CloudFormation. You can also learn how to use other AWS services that help you to monitor and secure your AWS CloudFormation and Cloud Control API resources.

Note the following areas where Cloud Control API differs from CloudFormation when addressing security and compliance concerns:

- For AWS Identity and Access Management (IAM) integration:
  - In IAM policies, Cloud Control API actions are specified with the `"cloudformation"` prefix.

    For example, the following policy grants create, read, update, and list (but not delete) resource actions.

```
{
    "Version":"2012-10-17",
    "Statement":[{
        "Effect":"Allow",
        "Action":[
            "cloudformation:CreateResource",
            "cloudformation:GetResource",
            "cloudformation:UpdateResource",
            "cloudformation:ListResources"
        ],
        "Resource":"*"
    }]
}
```

  - Cloud Control API does not currently support CloudFormation resource-level permissions.
  - Cloud Control API does not currently support use of CloudFormation conditions.

  For more information, see Controlling access with AWS Identity and Access Management in the *AWS CloudFormation User Guide*.
- Cloud Control API does not currently support VPC endpoints.
- Cloud Control API does not currently support Custom resources.

- When activity occurs in Cloud Control API and is recorded in AWS CloudTrail, the event source is listed as `cloudcontrolapi.amazonaws.com`.

  For more information, see Logging AWS CloudFormation API calls with AWS CloudTrail in the *AWS CloudFormation User Guide*.

Performing resource operations

Use AWS Cloud Control API to perform create, read, update, delete, and list (CRUD-L) operations on resources in your AWS account.

Contents

Prerequisites for using resources with Cloud Control API

To provision a specific resource using Cloud Control API, that resource type must support Cloud Control API and be available for use in your AWS account.

- **Resources that support Cloud Control API**

  For a list of resource types published by Amazon that support Cloud Control API, see Resource types that support Cloud Control API.

  Third-party resource types, both public and private, support Cloud Control API.

  For details about how to determine if a specific resource type supports Cloud Control API, see Determining if a resource type supports Cloud Control API.
- **Resources available for use in your AWS account**

  To be available for use in your account, public resource types must be activated, and private resource types must be registered. Resource types published by Amazon are public and activated by default. For more information, see Using resource types.

For information about using resource types, see Using resource types.

Specifying credentials for Cloud Control API

As part of performing operations on AWS resources on your behalf, Cloud Control API must make calls to the underlying AWS services that actually provision those resources. To do so, Cloud Control API requires the necessary credentials to access those services. There are two ways for you to enable Cloud Control API to acquire those credentials:

- **User credentials**

  By default, Cloud Control API creates a temporary session using your AWS user credentials, and uses that to make any necessary calls to downstream AWS services. This session lasts up to 24 hours, after which any remaining calls to AWS by Cloud Control API will fail.
- **Service role credentials**

  You can also specify a service role for Cloud Control API to assume during a resource operation, when you make the resource request. Among other advantages, specifying a service role enables Cloud Control API to make calls to underlying AWS services for up to 36 hours.

  To use a service role, specify the `RoleArn` parameter of the resource operation request.

  Because the Cloud Control API actions are part of the AWS CloudFormation service, the service role you specify is assumed by the CloudFormation service (`cloudformation.amazonaws.com`). For more information, see AWS CloudFormation service role in the *AWS CloudFormation User Guide*.

The permissions required for each resource handler are defined in the `handlers` section of that resource type's schema. For more information about viewing the resource schema, see Viewing resource type schemas. The `handlers` section is defined in the resource type definition schema.

Ensuring resource operation requests are unique when using Cloud Control API

As a best practice, we strongly recommend you specify an idempotency token with create, delete, and update resource operation requests. Preferably, specify a token that will be unique for every request, such as an universally unique identifier (UUID). Such a token ensures requests can be disambiguated in cases where a request must be retried.

The `create-resource`, `delete-resource`, and `update-resource` operations all take a `client-token` parameter, which can be set to an idempotency token.

Considerations when using Cloud Control API

We recommend that you take the following service behavior into account when performing resource operations using Cloud Control API:

- Cloud Control API performs each resource operation individually and independent of any other resource operations.
- A single resource operation request to Cloud Control API might actually consist of multiple calls to the underlying service that provisions the resource. Because of this, a resource request might fail when only partially completed, resulting in only some of the requested changes being applied to the resource.
- If a resource operation fails at any point, Cloud Control API does not roll back the resource to its previous state.
- You can only perform one resource operation at a time on a given resource using Cloud Control API. However, the resource can still be operated on directly, through the underlying service that provisioned it. We strongly recommend against this approach because it may lead to unpredictable behavior.

Creating a resource

Use the `create-resource` command to create a resource.

Composing the desired state of the resource

For Cloud Control API to create a resource, you must specify the *desired state* of the resource you want to create. The desired state consists of a listing of the resource properties you want to specify, and their desired values.

The properties of a resource are defined in its resource type schema. This includes whether the property is required, valid values, and other property constraints. For more information about viewing resource property definitions, see Viewing resource type schemas.

The desired state you specify must be valid against the resource type schema.

As an example, suppose you wanted to create an AWS::Logs::LogGroup resource with a specific name and a retention policy of 90 days. As a first step, you must compose the desired state of the resource, formatted as JSON text.

```
{
  "LogGroupName": "CloudApiLogGroup",
  "RetentionInDays": 90
}
```

When you call the `create-resource` command, you can pass the desired state directly inline as a string, or, for more complicated desired state definitions, specify a file location.

The following AWS Command Line Interface (AWS CLI) command creates the resource and specifies in the `desired-state` parameter that the `RetentionInDays` property of the resource is set to `90`, in addition to specifying the log group name.

```
aws cloudcontrol create-resource --type-name AWS::Logs::LogGroup --desired-state
 "{\"LogGroupName\": \"CloudApiLogGroup\",\"RetentionInDays\":90}"
```

Tracking the progress of a create resource request

The `create-resource` command returns a `ProgressEvent` object that you can use to track the current status of your resource create request. For more information, see Tracking the progress of resource requests.

Updating a resource

Use the `update-resource` command to make updates to an existing resource. This includes resources that were not originally provisioned using Cloud Control API.

> **Important**
> We strongly advise against using Cloud Control API to update resources that are under active management by other services. Doing so can lead to unexpected results. For example, do not use Cloud Control API to update resources that are currently part of an AWS CloudFormation stack.

To update an existing resource, you must specify the resource's identifier. For more information about determining a resource's identifier, see Identifying a resource.

Updating a resource entails changing resource property values. The properties of a resource are defined in its resource type schema. This includes whether the property is required, valid values, and other property constraints. For more information about viewing resource property definitions, see Viewing resource type schemas.

Composing the patch document

To update a resource, you first define the updates as a list of *patch operations* contained in a JSON patch document. This patch document must adhere to the standard defined in *RFC 6902 - JavaScript Object Notation (JSON) Patch*.

Each patch operation defines a single update to a specific resource property. The following properties are required:

- `op`: The operation type. Cloud Control API supports all operations defined in RFC 6902: `add`, `remove`, `replace`, `move`, `copy`, and `test`.
- `path`: The path to the resource property, relative to the `properties` section of the resource schema.

Depending on the operation, additional properties may be required. Refer to RFC 6902 for specifics.

When using the `update-resource` command, you can specify the patch document inline as a string, or specify a file location.

The following example updates the retention policy of an AWS::Logs::LogGroup resource named `CloudControlApiLogGroup`to 90 days.

```
aws cloudcontrol update-resource --type-name AWS::Logs::LogGroup --identifier
 CloudControlApiLogGroup --patch-document "[{\"op\":\"test\",\"path\":\"/RetentionInDays\",
\"value\":90}]"
```

How Cloud Control API updates resources

To update a resource, Cloud Control API first retrieves the current state of the resource and then updates the resource in a two-step process:

- Cloud Control API combines the patch operations specified in the update request with the current state of the resource, to generate the desired state of the resource after it's updated. Operations are applied sequentially in the order that they appear in the patch document. Each operation in the sequence is applied to the resource's current state; the resulting resource state becomes the target of the next operation.

  At this point, the entire update request fails if:
  - A patch operation included in the request is invalid.
  - A patch operation of `op` type `test` fails.

  In such cases, the entire update request fails and Cloud Control API makes no updates to the resource.
- Cloud Control API then calls the update handler of the resource type to update the resource.

  If the update handler fails at any point, *Cloud Control API does not roll back the resource to its previous state.*

For example, consider the following patch document that is defined to update an AWS::Logs::LogGroup resource. The document contains two patch operations. The first operation is of type `test` and checks to see if the resource's retention policy is set to 3653 days. If that is the case, the resource passes the test and Cloud Control API proceeds to the next operation. This operation replaces the current retention policy value with 180 days. If the resource's retention policy is set to a value of other than 3653 days, the first `test` operation fails and Cloud Control API never runs the second `replace` operation.

```
[
  {
    "op": "test",
    "path": "/RetentionInDays",
    "value":3653
  },
  {
    "op": "replace",
    "path": "/RetentionInDays",
    "value":180
  }
]
```

Tracking the progress of a update resource request

The `update-resource` command returns a `ProgressEvent` object that you can use to track the current status of your resource operation request. For more information, see Tracking the progress of resource requests.

Deleting a resource

Use the `delete-resource` command to delete an existing resource. You can delete the resource whether or not the resource was originally provisioned using Cloud Control API.

Important
We strongly advise against using Cloud Control API to delete resources that are under active management by other services. Doing socan lead to unexpected results. For example, do not use Cloud Control API to delete resources that are currently part of an AWS CloudFormation stack.

To update an existing resource, you must specify the resource's identifier. For more information about finding a resource's identifier, see Identifying a resource.

The following example deletes an AWS::Logs::LogGroup resource with the name of `CloudControlApiLogGroup`.

```
aws cloudcontrol delete-resource --type-name AWS::Logs::LogGroup --identifier
 CloudControlApiLogGroup
```

Tracking the progress of a delete resource request

The `delete-resource` command returns a `ProgressEvent` object that you can use to track the current status of your resource operation request. For more information, see Tracking the progress of resource requests.

Discovering resources

Use the `list-resources` command to discover the resources currently provisioned in your AWS account and AWS Region. This includes all resources of the specified resource type, regardless of whether they were provisioned through Cloud Control API, directly through the underlying service, or other mechanism (such as being part of an AWS CloudFormation stack).

The information returned for each resource includes:

- The resource's primary identifier.
- Optionally, it may include the *part or all* resource's properties, detailing the current state of the resource. For more information, see Viewing resource type schemas.

The following example returns a list of `AWS::Logs::LogGroup` resources.

```
aws cloudcontrol list-resources --type-name AWS::Logs::LogGroup
```

Cloud Control API returns a list of the resources in your account of the specified resource type. For the example above, `list-resources` returns the primary identifier and resource properties of all `AWS::Logs::LogGroup` resources in your account, regardless of whether they were provisioned by Cloud Control API. The returned information resembles the following, depending on the resources in your account.

```
{
    "TypeName": "AWS::Logs::LogGroup",
    "ResourceDescriptions": [
        {
            "Identifier": "CloudControlExample",
            "Properties": "{\"RetentionInDays\":180,\"LogGroupName\":\"CloudControlExample
\",\"Arn\":\"arn:aws:logs:us-west-2:090123456789:log-group:CloudControlExample:*\"}"
        },
        {
            "Identifier": "AnotherLogGroupResourceExample",
            "Properties": "{\"RetentionInDays\":90,\"LogGroupName\":
\"AnotherLogGroupResourceExample\",\"Arn\":\"arn:aws:logs:us-west-2:011111111111:log-
group:AnotherLogGroupResourceExample:*\"}"
        },
    ]
}
```

The following example requests a list of `AWS::Kinesis::Stream` resources.

```
aws cloudcontrol list-resources --type-name AWS::Kinesis::Stream
```

For Kinesis streams, Cloud Control API returns the primary identifier of each stream, along with a *subset* of the resource properties. In this case, just a single property, `Name`. You could then use a stream's primary identifier with `get-resource` to request the resource's full current state.

```
{
    "TypeName": "AWS::Kinesis::Stream",
    "ResourceDescriptions": [
        {
            "Identifier": "MyKinesisStream",
            "Properties": "{\"Name\":\"MyKinesisStream\"}"
        },
        {
            "Identifier": "AnotherStream",
            "Properties": "{\"Name\":\"AnotherStream\"}"
        }
    ]
}
```

Resources that require additional information

Certain resources require that you provide additional information about the resources that you want to list as part of your request. In these cases, you must use the `ResourceModel` parameter to specify these properties.

The table below lists these resources, and the properties you are required to specify in the `ResourceModel` parameter during list requests.

Resources	Required Properties
AWS::ApiGateway::DocumentationVersion	RestApiId
AWS::CloudFormation::ResourceVersion	TypeArn or TypeName
AWS::CustomerProfiles::Integration	DomainName
AWS::CustomerProfiles::ObjectType	DomainName
AWS::EC2::TransitGatewayMulticastGroupMember	TransitGatewayMulticastDomainId
AWS::EC2::TransitGatewayMulticastGroupSource	TransitGatewayMulticastDomainId
AWS::ECS::TaskSet	Cluster, Service, and Id
AWS::EKS::AddOn	ClusterName
AWS::EKS::FargateProfile	ClusterName
AWS::ElasticLoadBalancingV2::Listener	LoadBalancerArn
AWS::ElasticLoadBalancingV2::ListenerRule	ListenerArn
AWS::Glue::SchemaVersion	• SchemaDefinition, Schema/RegistryName, and Schema/SchemaName, or • SchemaDefinition and Schema/SchemaArn
AWS::Glue::SchemaVersionMetadata	SchemaVersionId
AWS::IoTSiteWise::AccessPolicy	• /AccessPolicyResource/Portal, or • /AccessPolicyResource/Project
AWS::IoTSiteWise::Dashboard	ProjectId
AWS::IoTSiteWise::Project	PortalId
AWS::Kendra::DataSource	IndexId
AWS::Kendra::Faq	IndexId
AWS::MediaConnect::FlowEntitlement	FlowArn
AWS::MediaConnect::FlowOutput	FlowArn
AWS::MediaConnect::FlowSource	FlowArn
AWS::MediaConnect::FlowVpcInterface	FlowArn
AWS::MediaPackage::Asset	PackagingGroupId
AWS::MediaPackage::PackagingConfiguration	PackagingGroupId

Resources	Required Properties
AWS::NetworkFirewall::LoggingConfiguration	• FirewallArn or • FirewallName
AWS::QuickSight::Analysis	AwsAccountId
AWS::QuickSight::Dashboard	AwsAccountId
AWS::QuickSight::DataSet	AwsAccountId
AWS::QuickSight::DataSource	AwsAccountId
AWS::QuickSight::Template	AwsAccountId
AWS::QuickSight::Theme	AwsAccountId
AWS::RDS::DBProxyTargetGroup	DBProxyName
AWS::S3Outposts::AccessPoint	Bucket
AWS::S3Outposts::Bucket	OutpostId
AWS::SSO::Assignment	InstanceArn, PermissionSetArn, PrincipalId, PrincipalType, TargetId, and TargetType
AWS::SSO::InstanceAccessControlAttributeConfiguration	InstanceArn
AWS::SSO::PermissionSet	InstanceArn and PermissionSetArn

Reading a resource's current state

Using a resource's primary identifier, you can call the `get-resource` command to retrieve detailed information about the resource. (For information about retrieving a resource's primary identifier, see Identifying a resource.)

The information returned by `get-resource` includes the resource's schema, which details the current state of the resource, including property values, supported events, and necessary permissions. For more information, see Viewing resource type schemas.

The following example returns the current state of an `AWS::Logs::LogGroup` resource named `LogGroupResourceExample`. For `AWS::Logs::LogGroup` resources, the name of a log group is its primary identifier.

```
aws cloudcontrol get-resource --type-name AWS::Logs::LogGroup --identifier
 LogGroupResourceExample
```

Managing resource operation requests

Because resource operations are asynchronous, resource requests such as `create-resource` and `update-resource` return a `ProgressEvent` object that contains information about the current state of your resource create or update request.

For example, a resource create request might initially return the following `ProgressEvent` object.

```
{
    "ProgressEvent": {
        "EventTime": "2021-08-09T18:17:15.219Z",
        "TypeName": "AWS::Logs::LogGroup",
        "OperationStatus": "IN_PROGRESS",
        "Operation": "CREATE",
        "Identifier": "LogGroupResourceExample",
        "RequestToken": "5f40c577-3534-4b20-9599-0b0123456789"
    }
}
```

The information returned in the `ProgressEvent` object includes a request token that you can then use to track or cancel a resource operation request.

> **Note**
> Resource operation requests expire after seven days.

Listing active resource operation requests

Use the `list-resource-requests` command to return a list of active resource operation requests for an AWS account and AWS Region. You can filter the list by request type and status.

Resource operation requests expire after seven days.

The following example returns active resource operation requests, but it filters out any resource create requests that are still in progress.

```
aws cloudcontrol list-resource-requests --resource-request-status-filter
 Operations=CREATE,OperationStatuses=IN_PROGRESS
```

The information returned for each resource operation includes a request token that you can then use to track or cancel a resource operation request.

```
{
    "ResourceRequestStatusSummaries": [
        {
            "EventTime": "2021-08-09T18:17:16.591Z",
            "TypeName": "AWS::Logs::LogGroup",
            "OperationStatus": "SUCCESS",
            "Operation": "CREATE",
            "Identifier": "LogGroupResourceExample",
            "RequestToken": "5f40c577-3534-4b20-9599-0b0123456789"
        }
    ]
}
```

Tracking the progress of resource operation requests

Use the `get-resource-request-status` command to track the progress of your resource operation request. This command takes the request token included in the `ProgressEvent` object generated during the initial resource operation request. (You can also retrieve the request token for a resource operation request using the `list-resource-requests` command.) The `get-resource-request-status` command returns an updated `ProgressEvent` object containing information on the current state of the request.

See the following example.

```
aws cloudcontrol get-resource-request-status --request-token
 5f40c577-3534-4b20-9599-0b0123456789
```

Canceling resource operation requests

Use the `cancel-resource-request` command to cancel a resource operation request that is currently in progress. Because you can only perform a single operation on a given resource at a time, there might be cases where you need to cancel the current resource operation to make the resource available so that another operation may be performed on it.

Canceling a resource request does not guarantee that Cloud Control API can immediately cancel all resource operations. Rather, Cloud Control API will stop making further calls to the resource event handler. A single resource operation request to Cloud Control API might actually consist of multiple calls to the underlying service that provisions the resource. Because of this, canceling a resource operation request might leave the request partially completed, resulting in only some of the requested changes being applied to the resource. Cloud Control API does not roll back the resource to its previous state.

Only resource operations requests with a status of `PENDING` or `IN_PROGRESS` can be cancelled.

Note
Although calling `CancelResourceRequest` cancels operations performed by Cloud Control API, it does not terminate any asynchronous operations that may have already started on downstream services.

Identifying resources

Every resource type has a property that is defined as its *primary identifier*. The value of this property must be unique for each resource of that type in a given AWS account and AWS Region. For example, many resource types include a `Name` property that must be unique for each resource of that type. In some cases, the primary identifier is defined as a combination of multiple properties that together form a unique identifier. By using this primary identifier, combined with the resource type, you can specify exactly which resource on which you want to perform resource operations such as `update-resource` or `delete-resource`.

In addition, some resource types define *secondary identifiers* that can also be used to uniquely identify resources of that type.

To determine which resource property (or combination of properties) is the primary identifier for a resource type, refer to the `primaryIdentifier` attribute of the resource type schema. The schema includes secondary identifiers defined, as well. For more information, see Viewing resource type schemas.

Getting a resource's primary identifier

You can find the identifier *value* for a specific resource by using Cloud Control API commands. Each of the following commands returns a `ProgressEvent` object that contains the primary identifier of the specified resources:

- `cancel-resource-request`
- `create-resource`
- `get-resource-request-status`
- `list-resource-requests`

Using a resource's primary identifier

When using Cloud Control API commands, you can specify the primary identifier or any secondary identifier defined for the resource type in its resource schema. You can only specify one identifier. Primary identifiers can be specified as a string or JSON; secondary identifiers must be specified as JSON.

For compound primary identifiers (that is, one that consists of multiple resource properties strung together), to specify the primary identifier as a string, list the property values *in the order that they are specified* in the primary identifier definition, separated by `|`.

For example, the primary identifier for the resource is defined as:

```
"primaryIdentifier": [ "/properties/DatabaseName", "/properties/TableName"
```

So, to specify the primary identifier of a resource as a string, you use the following format.

```
DatabaseName|TableName
```

For example, given a database with a database name of `MyDatabase` and table name of `MyTable`, you specify `MyDatabase|MyTable`.

For compound identifiers specified as JSON, property order is not required, as shown in the following example.

```
{"TableName": "MyTable", "DatabaseName": "MyDatabase"}
```

For more information about resource identifiers, see primaryidentifier in the *CloudFormation Command Line Interface User Guide for Extension Development*.

Using resource types

To use a resource type with AWS Cloud Control API, that resource type must be present and activated in your AWS account. Resource types published by Amazon are activated by default. You can choose to activate public resource types offered by third-party publishers as well. You do this, and other resource type management tasks, through the AWS CloudFormation extension registry.

The *extension registry* is a feature of AWS CloudFormation that contains detailed information about the resource types available for use in your account. These can include resource types published by third-parties, in addition to those published by Amazon. Using the registry, you can manage the resource types in your account, including:

- View the available and activated resource types.
- Register private resource types for use in your account.
- Activate public third-party resource types.
- Manage the resource type *versions*, including setting the default version of a resource type in your account.
- Set account-level configuration properties of a resource type, if it has any.

You can also use the AWS CloudFormation registry to view a resource type's schema, which contains important information about how to use the resource with Cloud Control API, such as property definitions and permission requirements. For more information, see Viewing resource type schemas.

The registry is available through the CloudFormation console, in addition to the CloudFormation API.

Note
Not all resource types listed in the CloudFormation registry currently support Cloud Control API. For more information, see Determining if a resource type supports Cloud Control API.

For more information about resource type management options, see Using the CloudFormation registry in the *AWS CloudFormation User Guide*.

Managing resource types using the AWS CloudFormation API

In addition to accessing the extension registry through the AWS CloudFormation console, you can use operations included in the AWS CloudFormation API to identify and manage the resource types in your account. The table below lists the API operations that you can use to discover, activate, and configure the resource types available in your account.

CloudFormation API operation	**AWS CLI command**	**Description**	
DescribeType	describe-type	Returns detailed information about a resource type.	
ListTypes	list-types	Returns summary information about a resource type.	
ActivateType	activate-type	Activates a public third-party resource type,	

CloudFormation API operation	AWS CLI command	Description	
		making it available for use in your account.	
DeactivateType	deactivate-type	Deactivates a public third-party resource type in your account.	
ListTypeVersions	list-type-versions	Returns summary information about the versions of a resource type.	
SetTypeDefaultVersion	set-type-default-version	Specifies the default version of a resource type.	
BatchDescribeTypeConfigurations	batch-describe-type-configurations	Returns configuration data for the specified resource types.	
SetTypeConfiguration	set-type-configuration	Specifies the configuration data for a resource type in your account.	
RegisterType	register-type	Registers a private third-party resource, making it available for use in your account.	
DeregisterType	deregister-type	Deregisters a private third-party resource, removing it from active use in your account.	

Determining if a resource type supports Cloud Control API

By default, resource types published in the CloudFormation registry automatically support Cloud Control API resource operations. This includes private resource types, in addition to public third-party resource types. However, the AWS CloudFormation registry also contains legacy resource types, classified as *non-provisionable*. These resource types don't currently support Cloud Control API, and you can't use them in resource operations.

For a list of the AWS public resource types that currently support Cloud Control API resource operations, see Resource types that support Cloud Control API.

You can also use the AWS Command Line Interface (AWS CLI) to generate a list of supported resource types or to determine if a specific resource type supports Cloud Control API.

Generating a list of supported resources using the AWS CLI

- Use the `list-types` command, with the following parameters:

- `type` – Specify `RESOURCE` to select only resource types.
- `visibility` – Specify `PUBLIC` to select public resources or `PRIVATE` for private resources.
- `provisioning-type` – Specify `FULLY_MUTABLE` or `IMMUTABLE` to select only those resource types that are provisionable.

For example, the following command selects the first 100 public resource types that are fully mutable from the CloudFormation registry.

```
aws cloudformation list-types --type RESOURCE --visibility PUBLIC --provisioning-type
 FULLY_MUTABLE --max-results 100
```

Determining if a specific resource type supports Cloud Control API using the AWS CLI

- Use the `describe-type` command to return details of the resource type.

  Resource types with a `ProvisioningType` of either `FULLY_MUTABLE` or `IMMUTABLE` support Cloud Control API resource operations.

  The following example returns details of the `AWS::Logs::LogGroup` resource type.

```
aws cloudformation describe-type --type RESOURCE --type-name "AWS::Logs::LogGroup"
```

Viewing resource type schemas

During resource create and update operations, you specify which resource properties to set and their values. The properties of a resource are defined in its *resource type schema*. This includes data type, whether the property is required, valid values, and other property constraints.

You can view a resource type's schema using the CloudFormation console or the AWS CLI. In addition, the *AWS CloudFormation User Guide* contains reference topics for each available resource type that AWS publishes. For detailed information about resource type properties, in addition to usage examples, see the corresponding topics in the AWS resource and property types reference section.

> **Note**
> Not all resource types listed in the *AWS CloudFormation User Guide* are available for use with Cloud Control API. To determine if a resource type is available, see Resource types that support Cloud Control API.

For detailed information about the resource type definition schema, which defines how resource type schema can be authored, see Resource type definition schema in the *CloudFormation CLI User Guide for Extension Development*.

For information about how to view an existing resource's current state, which includes its current property values, see Reading resources.

Viewing a resource type schema using the AWS CloudFormation console

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation.
2. In the **CloudFormation** navigation pane, under **Registry**, select **Activate extensions**.
3. On the **Resource types** tab, select the resource type that you want to view the resource schema of.

CloudFormation displays the resource type details page. The resource schema is displayed on the **Schema** tab.

Viewing a resource type schema using the AWS CLI

- Run `describe-type`.

  In the returned output, the `Schema` structure contains the resource type schema.

  For example, the following command returns information about the `AWS::Logs::LogGroup` resource type.

```
aws cloudformation describe-type --type RESOURCE --type-name AWS::Logs::LogGroup
```

Viewing resource property attributes

Resource type properties are defined in the `properties` section of the resource type schema. This includes the property data type, whether the property is required, and any constraints such as allowable values or required patterns.

In addition, certain attributes set at the resource level govern when or if a property can be specified. This includes:

- Properties defined as `required` must be specified in the desired state during resource creation.
- Properties defined as `createOnlyProperties` can be set by users, but only during resource creation.
- Properties defined as `readOnlyProperties` can't be set by users.
- Properties defined as `writeOnlyProperties` can be specified by users when creating or updating a resource but can't be returned during a read or list request.

Viewing supported resource operations

You can determine which operations a resource type supports by referring to the `handlers` section of its resource type schema. If the resource type supports an operation, it's listed in the `handlers` section, and it contains a `permissions` element that lists the permissions that the handler requires.

For example, below is the `handlers` section of the resource type schema for the `AWS::Logs::LogGroup` resource type. This section shows that this resource type supports all five resource operations, and lists the permissions that each handler requires.

```
"handlers": {
  "create": {
    "permissions": [
      "logs:DescribeLogGroups",
      "logs:CreateLogGroup",
      "logs:PutRetentionPolicy"
    ]
  },
  "read": {
    "permissions": [
      "logs:DescribeLogGroups"
    ]
  },
  "update": {
    "permissions": [
        "logs:DescribeLogGroups",
        "logs:AssociateKmsKey",
        "logs:DisassociateKmsKey",
        "logs:PutRetentionPolicy",
        "logs:DeleteRetentionPolicy"
      ]
    },
    "delete": {
      "permissions": [
        "logs:DescribeLogGroups",
        "logs:DeleteLogGroup"
      ]
    },
    "list": {
      "permissions": [
        "logs:DescribeLogGroups"
      ]
    }
  }
```

Resource types that support Cloud Control API

The following table lists the public resource types published by Amazon that currently support AWS Cloud Control API, organized by service. Each resource type name links to the corresponding reference topic for that resource type in the Resource and property types reference section of the *AWS CloudFormation User Guide*.

Third-party resource types, both public and private, support Cloud Control API.

For information about how to determine if a specific resource type supports Cloud Control API, see Determining if a resource type supports Cloud Control API. For more information about using resource types, see Using resource types.

Table updated: September 30, 2021.

Service	Resource
AWS Amplify	AWS::Amplify::App AWS::Amplify::Branch AWS::Amplify::Domain
Amazon API Gateway	AWS::ApiGateway::Account AWS::ApiGateway::ApiKey AWS::ApiGateway::ClientCertificate AWS::ApiGateway::DocumentationVersion AWS::ApiGateway::DomainName AWS::ApiGateway::Model AWS::ApiGateway::RequestValidator AWS::ApiGateway::Resource AWS::ApiGateway::UsagePlanKey AWS::ApiGateway::UsagePlan
Amazon AppFlow	AWS::AppFlow::ConnectorProfile AWS::AppFlow::Flow
AWS App Runner	AWS::AppRunner::Service
Amazon Managed Service for Prometheus	AWS::APS::Workspace
Amazon Athena	AWS::Athena::DataCatalog AWS::Athena::NamedQuery

Service	Resource
	AWS::Athena::PreparedStatement AWS::Athena::WorkGroup
AWS Audit Manager	AWS::AuditManager::Assessment
AWS Backup	AWS::Backup::BackupPlan AWS::Backup::BackupSelection AWS::Backup::BackupVault
AWS Budgets	AWS::Budgets::BudgetsAction
AWS Billing and Cost Management	AWS::CE::AnomalyMonitor AWS::CE::AnomalySubscription AWS::CE::CostCategory AWS::CUR::ReportDefinition
AWS Certificate Manager	AWS::ACMPCA::Certificate AWS::ACMPCA::CertificateAuthority AWS::ACMPCA::CertificateAuthorityActivation AWS::ACMPCA::Permission AWS::CertificateManager::Account
AWS Chatbot	AWS::Chatbot::SlackChannelConfiguration
AWS CloudFormation	AWS::CloudFormation::ModuleDefaultVersion AWS::CloudFormation::ModuleVersion AWS::CloudFormation::PublicTypeVersion AWS::CloudFormation::Publisher AWS::CloudFormation::ResourceDefaultVersion AWS::CloudFormation::ResourceVersion AWS::CloudFormation::StackSet AWS::CloudFormation::TypeActivation

Service	Resource
Amazon CloudFront	AWS::CloudFront::CachePolicy AWS::CloudFront::CloudFrontOriginAccessIdentity AWS::CloudFront::Distribution AWS::CloudFront::Function AWS::CloudFront::KeyGroup AWS::CloudFront::OriginRequestPolicy AWS::CloudFront::PublicKey AWS::CloudFront::RealtimeLogConfig
AWS CloudTrail	AWS::CloudTrail::Trail
Amazon CloudWatch	AWS::ApplicationInsights::Application AWS::CloudWatch::CompositeAlarm AWS::CloudWatch::MetricStream AWS::Logs::LogGroup AWS::Logs::QueryDefinition AWS::Logs::ResourcePolicy AWS::Synthetics::Canary
AWS CodeArtifact	AWS::CodeArtifact::Domain AWS::CodeArtifact::Repository
Amazon CodeGuru Profiler	AWS::CodeGuruProfiler::ProfilingGroup
Amazon CodeGuru Reviewer	AWS::CodeGuruReviewer::RepositoryAssociation
AWS CodeStar	AWS::CodeStarConnections::Connection AWS::CodeStarNotifications::NotificationRule
Amazon Connect	AWS::AppIntegrations::EventIntegration AWS::Connect::QuickConnect AWS::CustomerProfiles::Domain AWS::CustomerProfiles::Integration AWS::CustomerProfiles::ObjectType

Service	Resource
AWS Config	AWS::Config::ConfigurationAggregator AWS::Config::ConformancePack AWS::Config::OrganizationConformancePack AWS::Config::StoredQuery
AWS DataSync	AWS::DataSync::Agent AWS::DataSync::LocationEFS AWS::DataSync::LocationFSxWindows AWS::DataSync::LocationNFS AWS::DataSync::LocationObjectStorage AWS::DataSync::LocationS3 AWS::DataSync::LocationSMB AWS::DataSync::Task
Amazon Detective	AWS::Detective::Graph AWS::Detective::MemberInvitation
Amazon DevOps Guru	AWS::DevOpsGuru::NotificationChannel AWS::DevOpsGuru::ResourceCollection
Amazon DynamoDB	AWS::DynamoDB::GlobalTable

Service	Resource
Amazon EC2	AWS::EC2::CarrierGateway AWS::EC2::DHCPOptions AWS::EC2::EC2Fleet AWS::EC2::EgressOnlyInternetGateway AWS::EC2::EnclaveCertificateIamRoleAssociation AWS::EC2::FlowLog AWS::EC2::GatewayRouteTableAssociation AWS::EC2::LocalGatewayRoute AWS::EC2::LocalGatewayRouteTableVPCAssociation AWS::EC2::NetworkInsightsAnalysis AWS::EC2::NetworkInsightsPath AWS::EC2::PrefixList AWS::EC2::SpotFleet AWS::EC2::Subnet AWS::EC2::TransitGateway AWS::EC2::TransitGatewayConnect AWS::EC2::TransitGatewayMulticastDomain AWS::EC2::TransitGatewayMulticastGroupMember AWS::EC2::TransitGatewayMulticastGroupSource AWS::EC2::TransitGatewayPeeringAttachment AWS::EC2::TransitGatewayVpcAttachment
Amazon EC2 Auto Scaling	AWS::AutoScaling::WarmPool
EC2 Image Builder	AWS::ImageBuilder::Component AWS::ImageBuilder::ContainerRecipe AWS::ImageBuilder::DistributionConfiguration AWS::ImageBuilder::Image AWS::ImageBuilder::ImagePipeline AWS::ImageBuilder::ImageRecipe AWS::ImageBuilder::InfrastructureConfiguration

Service	Resource
Amazon ECR	AWS::ECR::PublicRepository AWS::ECR::RegistryPolicy AWS::ECR::ReplicationConfiguration AWS::ECR::Repository
Amazon ECS	AWS::ECS::CapacityProvider AWS::ECS::Cluster AWS::ECS::ClusterCapacityProviderAssociations AWS::ECS::PrimaryTaskSet AWS::ECS::Service AWS::ECS::TaskDefinition AWS::ECS::TaskSet
Amazon EFS	AWS::EFS::AccessPoint AWS::EFS::FileSystem AWS::EFS::MountTarget
Amazon EKS	AWS::EKS::AddOn AWS::EKS::FargateProfile
Amazon ElastiCache	AWS::ElastiCache::GlobalReplicationGroup AWS::ElastiCache::User AWS::ElastiCache::UserGroup
Elastic Load Balancing	AWS::ElasticLoadBalancingV2::Listener AWS::ElasticLoadBalancingV2::ListenerRule
Amazon EMR	AWS::EMR::Studio AWS::EMR::StudioSessionMapping
Amazon EMR	AWS::EMRContainers::VirtualCluster
Amazon EventBridge	AWS::Events::ApiDestination AWS::Events::Archive AWS::Events::Connection AWS::EventSchemas::RegistryPolicy
Amazon FinSpace	AWS::FinSpace::Environment

Service	Resource
AWS Firewall Manager	AWS::FMS::NotificationChannel AWS::FMS::Policy
Amazon Fraud Detector	AWS::FraudDetector::Detector AWS::FraudDetector::EntityType AWS::FraudDetector::EventType AWS::FraudDetector::Label AWS::FraudDetector::Outcome AWS::FraudDetector::Variable
Amazon GameLift	AWS::GameLift::Alias AWS::GameLift::Fleet AWS::GameLift::GameServerGroup
AWS Global Accelerator	AWS::GlobalAccelerator::Accelerator AWS::GlobalAccelerator::EndpointGroup AWS::GlobalAccelerator::Listener
AWS Glue	AWS::Glue::Registry AWS::Glue::Schema AWS::Glue::SchemaVersion AWS::Glue::SchemaVersionMetadata
AWS Glue DataBrew	AWS::DataBrew::Dataset AWS::DataBrew::Job AWS::DataBrew::Project AWS::DataBrew::Recipe AWS::DataBrew::Schedule
AWS FIS	AWS::FIS::ExperimentTemplate
Amazon HealthLake	AWS::HealthLake::FHIRDatastore
AWS Ground Station	AWS::GroundStation::Config AWS::GroundStation::DataflowEndpointGroup AWS::GroundStation::MissionProfile

Service	Resource
AWS Identity and Access Management	AWS::AccessAnalyzer::Analyzer AWS::IAM::OIDCProvider AWS::IAM::SAMLProvider AWS::IAM::ServerCertificate AWS::IAM::VirtualMFADevice
AWS IoT Core	AWS::IoT::AccountAuditConfiguration AWS::IoT::Authorizer AWS::IoT::Certificate AWS::IoT::CustomMetric AWS::IoT::Dimension AWS::IoT::DomainConfiguration AWS::IoT::FleetMetric AWS::IoT::MitigationAction AWS::IoT::ProvisioningTemplate AWS::IoT::ScheduledAudit AWS::IoT::SecurityProfile AWS::IoT::TopicRule AWS::IoT::TopicRuleDestination AWS::IoTCoreDeviceAdvisor::SuiteDefinition AWS::IoTFleetHub::Application AWS::IoTWireless::Destination AWS::IoTWireless::DeviceProfile AWS::IoTWireless::PartnerAccount AWS::IoTWireless::ServiceProfile AWS::IoTWireless::TaskDefinition AWS::IoTWireless::WirelessDevice AWS::IoTWireless::WirelessGateway
AWS IoT Events	AWS::IoTEvents::DetectorModel AWS::IoTEvents::Input
AWS IoT Greengrass	AWS::GreengrassV2::ComponentVersion

Service	Resource
AWS IoT SiteWise	AWS::IoTSiteWise::AccessPolicy AWS::IoTSiteWise::Asset AWS::IoTSiteWise::AssetModel AWS::IoTSiteWise::Dashboard AWS::IoTSiteWise::Gateway AWS::IoTSiteWise::Portal AWS::IoTSiteWise::Project
Amazon Interactive Video Service	AWS::IVS::Channel AWS::IVS::PlaybackKeyPair AWS::IVS::RecordingConfiguration AWS::IVS::StreamKey
Amazon Kendra	AWS::Kendra::DataSource AWS::Kendra::Faq AWS::Kendra::Index
Amazon Keyspaces (for Apache Cassandra)	AWS::Cassandra::Keyspace AWS::Cassandra::Table
Amazon Kinesis	AWS::Kinesis::Stream
Amazon Kinesis Data Firehose	AWS::KinesisFirehose::DeliveryStream
AWS Key Management Service	AWS::KMS::Alias AWS::KMS::Key AWS::KMS::ReplicaKey
AWS Lambda	AWS::Lambda::CodeSigningConfig AWS::Lambda::EventSourceMapping AWS::Lambda::Function
AWS License Manager	AWS::LicenseManager::Grant AWS::LicenseManager::License

Service	Resource
Amazon Location Service	AWS::Location::GeofenceCollection AWS::Location::Map AWS::Location::PlaceIndex AWS::Location::RouteCalculator AWS::Location::Tracker AWS::Location::TrackerConsumer
Amazon Lookout for Equipment	AWS::LookoutEquipment::InferenceScheduler
Amazon Lookout for Metrics	AWS::LookoutMetrics::Alert AWS::LookoutMetrics::AnomalyDetector
Amazon Lookout for Vision	AWS::LookoutVision::Project
Amazon Macie	AWS::Macie::CustomDataIdentifier AWS::Macie::FindingsFilter AWS::Macie::Session
AWS Elemental MediaConnect	AWS::MediaConnect::Flow AWS::MediaConnect::FlowEntitlement AWS::MediaConnect::FlowOutput AWS::MediaConnect::FlowSource AWS::MediaConnect::FlowVpcInterface
AWS Elemental MediaPackage	AWS::MediaPackage::Asset AWS::MediaPackage::Channel AWS::MediaPackage::OriginEndpoint AWS::MediaPackage::PackagingConfiguration AWS::MediaPackage::PackagingGroup
Amazon Managed Workflows for Apache Airflow (Amazon MWAA)	AWS::MWAA::Environment
AWS Network Firewall	AWS::NetworkFirewall::Firewall AWS::NetworkFirewall::FirewallPolicy AWS::NetworkFirewall::LoggingConfiguration AWS::NetworkFirewall::RuleGroup

Service	Resource
Transit Gateway Network Manager	AWS::NetworkManager::CustomerGatewayAssociation AWS::NetworkManager::Device AWS::NetworkManager::GlobalNetwork AWS::NetworkManager::Link AWS::NetworkManager::LinkAssociation AWS::NetworkManager::Site AWS::NetworkManager::TransitGatewayRegistration
Amazon Nimble Studio	AWS::NimbleStudio::LaunchProfile AWS::NimbleStudio::StreamingImage AWS::NimbleStudio::Studio AWS::NimbleStudio::StudioComponent
Amazon QLDB	AWS::QLDB::Stream
Amazon OpenSearch Service	AWS::OpenSearchService::Domain
AWS OpsWorks CM	AWS::OpsWorksCM::Server
Amazon QuickSight	AWS::QuickSight::Analysis AWS::QuickSight::Dashboard AWS::QuickSight::DataSet AWS::QuickSight::DataSource AWS::QuickSight::Template AWS::QuickSight::Theme
AWS Resource Groups	AWS::ResourceGroups::Group
Amazon Relational Database Service	AWS::RDS::DBProxy AWS::RDS::DBProxyEndpoint AWS::RDS::DBProxyTargetGroup AWS::RDS::GlobalCluster
Amazon Redshift	AWS::Redshift::Cluster

Service	Resource
AWS RoboMaker	AWS::RoboMaker::Fleet AWS::RoboMaker::Robot AWS::RoboMaker::RobotApplicationVersion AWS::RoboMaker::SimulationApplication AWS::RoboMaker::SimulationApplicationVersion
Amazon Route 53	AWS::Route53::DNSSEC AWS::Route53::HealthCheck AWS::Route53::HostedZone AWS::Route53::KeySigningKey AWS::Route53RecoveryControl::Cluster AWS::Route53RecoveryControl::ControlPanel AWS::Route53RecoveryControl::RoutingControl AWS::Route53RecoveryControl::SafetyRule AWS::Route53RecoveryReadiness::Cell AWS::Route53RecoveryReadiness::ReadinessCheck AWS::Route53RecoveryReadiness::RecoveryGroup AWS::Route53RecoveryReadiness::ResourceSet AWS::Route53Resolver::FirewallDomainList AWS::Route53Resolver::FirewallRuleGroup AWS::Route53Resolver::FirewallRuleGroupAssociation AWS::Route53Resolver::ResolverDNSSECConfig AWS::Route53Resolver::ResolverQueryLoggingConfig AWS::Route53Resolver::ResolverQueryLoggingConfigAssociation

Service	Resource
Amazon Simple Storage Service	AWS::S3::AccessPoint AWS::S3::MultiRegionAccessPoint AWS::S3::MultiRegionAccessPointPolicy AWS::S3::StorageLens AWS::S3ObjectLambda::AccessPoint AWS::S3ObjectLambda::AccessPointPolicy AWS::S3Outposts::AccessPoint AWS::S3Outposts::Bucket AWS::S3Outposts::BucketPolicy AWS::S3Outposts::Endpoint
Amazon SageMaker	AWS::SageMaker::App AWS::SageMaker::AppImageConfig AWS::SageMaker::DataQualityJobDefinition AWS::SageMaker::Device AWS::SageMaker::DeviceFleet AWS::SageMaker::Domain AWS::SageMaker::FeatureGroup AWS::SageMaker::Image AWS::SageMaker::ImageVersion AWS::SageMaker::ModelBiasJobDefinition AWS::SageMaker::ModelExplainabilityJobDefinition AWS::SageMaker::ModelQualityJobDefinition AWS::SageMaker::ModelPackageGroup AWS::SageMaker::MonitoringSchedule AWS::SageMaker::Pipeline AWS::SageMaker::Project AWS::SageMaker::UserProfile

Service	Resource
AWS Service Catalog	AWS::ServiceCatalog::CloudFormationProvisionedProduct AWS::ServiceCatalog::ServiceAction AWS::ServiceCatalog::ServiceActionAssociation AWS::ServiceCatalogAppRegistry::Application AWS::ServiceCatalogAppRegistry::AttributeGroup AWS::ServiceCatalogAppRegistry::AttributeGroupAssociation AWS::ServiceCatalogAppRegistry::ResourceAssociation
AWS Signer	AWS::Signer::ProfilePermission AWS::Signer::SigningProfile
Amazon Simple Email Service	AWS::SES::ConfigurationSet AWS::SES::ContactList
AWS Single Sign-On	AWS::SSO::Assignment AWS::SSO::InstanceAccessControlAttributeConfiguration AWS::SSO::PermissionSet
AWS Step Functions	AWS::StepFunctions::StateMachine
AWS Systems Manager	AWS::SSM::Association AWS::SSM::Document AWS::SSM::ResourceDataSync AWS::SSMContacts::Contact AWS::SSMContacts::ContactChannel AWS::SSMIncidents::ReplicationSet AWS::SSMIncidents::ResponsePlan
Amazon Timestream	AWS::Timestream::Database AWS::Timestream::Table
AWS WAF	AWS::WAFv2::IPSet AWS::WAFv2::LoggingConfiguration AWS::WAFv2::RegexPatternSet AWS::WAFv2::RuleGroup AWS::WAFv2::WebACL AWS::WAFv2::WebACLAssociation

Service	Resource
AWS X-Ray	AWS::XRay::Group AWS::XRay::SamplingRule

Document history for the Cloud Control API User Guide

The following table describes the documentation releases for AWS Cloud Control API.

update-history-change	update-history-description	update-history-date
Initial release (p. 43)	Initial release of the Cloud Control API User Guide	September 30, 2021