Amazon S3 Examples

Amazon Simple Storage Service (Amazon S3) is a web service that provides highly scalable cloud storage. Amazon S3 provides easy to use object storage, with a simple web service interface to store and get any amount of data from anywhere on the web.

You can use the following examples to access Amazon Simple Storage Service (Amazon S3) using AWS Python SDK. For more information about Amazon S3, see the Amazon S3 documentation.

**Examples**

* Creating and Using Amazon S3 Buckets
* Downloading a File from an S3 Bucket
* Configuring Amazon S3 Buckets
* Managing Amazon S3 Bucket Access Permissions
* Working with Amazon S3 Bucket Policies
* Using an Amazon S3 Bucket as a Static Web Host

# Creating and Using Amazon S3 Buckets

This Python example shows you how to:

* Obtain and display a list of Amazon S3 buckets in your account.
* Create an Amazon S3 bucket.
* Upload an object to a specified bucket.

## The Scenario

In this example, Python code is used to obtain a list of existing Amazon S3 buckets, create a bucket, and upload a file to a specified bucket. The code uses the AWS SDK for Python to get information from and upload files to an Amazon S3 bucket using these methods of the Amazon S3 client class:

* list\_buckets : show list of existing buckets
* create\_bucket: create new bucket
* upload\_file : send or load local file to s3

All the example code for the Amazon Web Services (AWS) SDK for Python is available here on GitHub.

## Prerequisite Tasks

To set up and run this example, you must first complete this task:

* Configure your AWS credentials, as described in Quickstart.

## Display a List of Amazon S3 Buckets

List all the buckets owned by the authenticated sender of the request.

The example below shows how to:

* List buckets using list\_buckets.

### Example

**import** **boto3**

*# Create an S3 client*

#s3 **=** boto3**.**client('s3')

s3 = boto3.client('s3',aws\_access\_key\_id='ACCESS\_KEY',aws\_secret\_access\_key='SECRET\_KEY')

#response = s3.get\_object(Bucket='BUCKET', Key='KEY')

*# Call S3 to list current buckets*

response **=** s3**.**list\_buckets()

*# Get a list of all bucket names from the response*

buckets **=** [bucket['Name'] **for** bucket **in** response['Buckets']]

*# Print out the bucket list*

**print**("Bucket List: *%s*" **%** buckets)

## Create an Amazon S3 Bucket

The example below shows how to:

* Create a new bucket using create\_bucket.

### Example

**import** **boto3**

s3 **=** boto3**.**client('s3')

s3**.**create\_bucket(Bucket**=**'my-bucket')

## Upload a File to an Amazon S3 Bucket

The example below shows how to:

* Upload a file to a bucket using upload\_file.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

filename **=** 'file.txt'

bucket\_name **=** 'my-bucket'

*# Uploads the given file using a managed uploader, which will split up large*

*# files automatically and upload parts in parallel.*

s3**.**upload\_file(filename, bucket\_name, filename)

# Downloading a File from an S3 Bucket

This example shows how to download a file from an S3 bucket, using **S3.Bucket.download\_file()**.

## Prerequisites

To set up and run this example, you must first:

* Configure your AWS credentials, as described in Quickstart.
* Create an S3 bucket and upload a file to the bucket.
* Replace the BUCKET\_NAME and KEY values in the code snippet with the name of your bucket and the key for the uploaded file.

## Downloading a File

The example below tries to download an S3 object to a file. If the service returns a 404 error, it prints an error message indicating that the object doesn't exist.

**import** **boto3**

**import** **botocore**

BUCKET\_NAME **=** 'my-bucket' *# replace with your bucket name*

KEY **=** 'my\_image\_in\_s3.jpg' *# replace with your object key*

s3 **=** boto3**.**resource('s3')

**try**:

s3**.**Bucket(BUCKET\_NAME)**.**download\_file(KEY, 'my\_local\_image.jpg')

**except** botocore**.**exceptions**.**ClientError **as** e:

**if** e**.**response['Error']['Code'] **==** "404":

**print**("The object does not exist.")

**else**:

**raise**

# Configuring Amazon S3 Buckets

This Python example shows you how to configure the cross-origin resource sharing (CORS) permissions for a bucket.

## The Scenario

In this example, Python code is used to list your Amazon S3 buckets and to configure CORS and bucket logging. The Python code uses the AWS SDK for Python to configure a selected Amazon S3 bucket using these methods of the Amazon S3 client class:

* get\_bucket\_cors
* put\_bucket\_cors.

For more information about using CORS configuration with an Amazon S3 bucket, see Cross-Origin Resource Sharing (CORS) in the Amazon Simple Storage Service Developer Guide.

All the example code for the Amazon Web Services (AWS) SDK for Python is available here on GitHub.

## Prerequisite Tasks

To set up and run this example, you must first complete this task:

* Configure your AWS credentials, as described in Quickstart.

## Get a Bucket CORS Configuration

The example below shows how to:

* Get a CORS configuration for a specified bucket using get\_bucket\_cors.

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call S3 to get CORS configuration for selected bucket*

result **=** s3**.**get\_bucket\_cors(Bucket**=**'my-bucket')

## Set a Bucket CORS Configuration

The example below shows how to:

* Set a CORS configuration for a specified bucket using put\_bucket\_cors.

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Create the CORS configuration*

cors\_configuration **=** {

'CORSRules': [{

'AllowedHeaders': ['Authorization'],

'AllowedMethods': ['GET', 'PUT'],

'AllowedOrigins': ['\*'],

'ExposeHeaders': ['GET', 'PUT'],

'MaxAgeSeconds': 3000

}]

}

*# Set the new CORS configuration on the selected bucket*

s3**.**put\_bucket\_cors(Bucket**=**'my-bucket', CORSConfiguration**=**cors\_configuration)

Downloading a File from an S3 Bucket

Managing Amazon S3 Bucket Access Permissions

# Managing Amazon S3 Bucket Access Permissions

This Python example shows you how to get or set the access control list for an Amazon S3 bucket.

## The Scenario

In this example, a Python code is used to display the bucket access control list (ACL) for a selected bucket. The code uses the AWS SDK for Python to manage Amazon S3 bucket access permissions using this method of the Amazon S3 client class:

* get\_bucket\_acl.

For more information about access control lists for Amazon S3 buckets, see Managing Access with ACLs in the Amazon Simple Storage Service Developer Guide.

All the example code for the Amazon Web Services (AWS) SDK for Python is available here on GitHub.

## Prerequisite Tasks

To set up and run this example, you must first complete this task:

* Configure your AWS credentials, as described in Quickstart.

## Get a Specified Bucket Access Control (ACL) List

Access control lists (ACLs) are one of the resource-based access policy option you can use to manage access to your buckets and objects. You can use ACLs to grant basic read/write permissions to other AWS accounts.

The example below shows how to:

* Get the bucket ACL for a specified bucket using get\_bucket\_acl.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call to S3 to retrieve the policy for the given bucket*

result **=** s3**.**get\_bucket\_acl(Bucket**=**'my-bucket')

**print**(result)

# Working with Amazon S3 Bucket Policies

This Python example shows you how to:

* Get the bucket policy of an Amazon S3 bucket.
* Add or update the bucket policy of an Amazon S3 bucket.
* Delete the bucket policy of an Amazon S3 bucket.

## The Scenario

In this example, Python code is used to get, set, or delete a bucket policy on an Amazon S3 bucket. The code uses the AWS SDK for Python to configure policy for a selected Amazon S3 bucket using these methods of the Amazon S3 client class:

* get\_bucket\_policy.
* put\_bucket\_policy.
* delete\_bucket\_policy

For more information about bucket policies for Amazon S3 buckets, see Using Bucket Policies and User Policies in the Amazon Simple Storage Service Developer Guide.

All the example code for the Amazon Web Services (AWS) SDK for Python is available here on GitHub.

## Prerequisite Tasks

To set up and run this example, you must first complete this task:

* Configure your AWS credentials, as described in Quickstart.

## Get the Current Bucket Policy

The example below shows how to:

* Get the bucket ACL for a specified bucket using get\_bucket\_policy.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call to S3 to retrieve the policy for the given bucket*

result **=** s3**.**get\_bucket\_policy(Bucket**=**'my-bucket')

**print**(result)

## Set a Simple Bucket Policy

The example below shows how to:

* Set the bucket policy for a specified bucket using put\_bucket\_policy.

### Example

**import** **boto3**

**import** **json**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

bucket\_name **=** 'my-bucket'

*# Create the bucket policy*

bucket\_policy **=** {

'Version': '2012-10-17',

'Statement': [{

'Sid': 'AddPerm',

'Effect': 'Allow',

'Principal': '\*',

'Action': ['s3:GetObject'],

'Resource': "arn:aws:s3:::*%s*/\*" **%** bucket\_name

}]

}

*# Convert the policy to a JSON string*

bucket\_policy **=** json**.**dumps(bucket\_policy)

*# Set the new policy on the given bucket*

s3**.**put\_bucket\_policy(Bucket**=**bucket\_name, Policy**=**bucket\_policy)

## Delete a Bucket Policy

The example below shows how to:

* Delete a bucket policy for a specified bucket using delete\_bucket\_policy.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call S3 to delete the policy for the given bucket*

s3**.**delete\_bucket\_policy(Bucket**=**'my-bucket')

# Using an Amazon S3 Bucket as a Static Web Host

This Python example shows you how to set up an Amazon S3 bucket as a static web host. The Scenario

In this example, Python code is used to configure any of your buckets to act as a static web host. The code uses the AWS SDK for Python to configure a selected Amazon S3 bucket using these methods of the Amazon S3 client class:

* get\_bucket\_website
* put\_bucket\_website
* delete\_bucket\_website

For more information about using an Amazon S3 bucket as a static web host, see Hosting a Static Website on Amazon S3 in the Amazon Simple Storage Service Developer Guide.

All the example code for the Amazon Web Services (AWS) SDK for Python is available here on GitHub.

## Prerequisite Tasks

To set up and run this example, you must first complete this task:

* Configure your AWS credentials, as described in Quickstart.

## Get the Current Bucket Website Configuration

The example below shows how to:

* Get the bucket website configuration using get\_bucket\_website.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call to S3 to retrieve the policy for the given bucket*

result **=** s3**.**get\_bucket\_website(Bucket**=**'my-bucket')

## Set a Bucket Website Configuration

The example below shows how to:

* Set a bucket website configuration using put\_bucket\_website.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Create the configuration for the website*

website\_configuration **=** {

'ErrorDocument': {'Key': 'error.html'},

'IndexDocument': {'Suffix': 'index.html'},

}

*# Set the new policy on the selected bucket*

s3**.**put\_bucket\_website(

Bucket**=**'my-bucket',

WebsiteConfiguration**=**website\_configuration

)

## Delete a Bucket Website Configuration

The example below shows how to:

* Delete a bucket website configuration using delete\_bucket\_website.

### Example

**import** **boto3**

*# Create an S3 client*

s3 **=** boto3**.**client('s3')

*# Call S3 to delete the website policy for the given bucket*

s3**.**delete\_bucket\_website(Bucket**=**'my-bucket')