Using Open Data with Amazon S3

**SPL-60 Version 2.2.10**

© 2023 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. All trademarks are the property of their owners.

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at [*AWS Training and Certification*](https://support.aws.amazon.com/#/contacts/aws-training).

**Lab overview**

This lab demonstrates the steps to upload data to Amazon Simple Storage Service (Amazon S3) and make it available for anyone to access via a web browser. You learn to create an Amazon S3 bucket, configure it to host a static website, upload objects to it, and even use JavaScript to display those objects on a web page. Along the way, you learn some best practices for publishing data on Amazon S3. At the end of this lab you have deployed a simple web site that makes data easy to access and provides basic documentation of the data.

OBJECTIVES

When you finish the lab, you will be able to:

* Create an Amazon S3 bucket.
* Upload objects to Amazon S3.
* Create a Bucket Policy to set permissions on a bucket.
* Create a Cross-Origin Resource Sharing (CORS) Configuration policy for a bucket.
* Configure a bucket for static website hosting.
* Use JavaScript to display the contents of a bucket on a static webpage.

DURATION

This lab requires approximately *45* minutes to complete.

ICON KEY

Various icons are used throughout this lab to call attention to different types of instructions and notes. The following list explains the purpose for each icon:

* **Expected output:** A sample output that you can use to verify the output of a command or edited file.
* **Note:** A hint, tip, or important guidance.
* **Learn more:** Where to find more information.
* **Caution:** Information of special interest or importance (not so important to cause problems with the equipment or data if you miss it, but it could result in the need to repeat certain steps).
* **Refresh:** A time when you might need to refresh a web browser page or list to show new information.
* **Copy edit:** A time when copying a command, script, or other text to a text editor (to edit specific variables within it) might be easier than editing directly in the command line or terminal.
* **Security:** An opportunity to incorporate security best practices.
* **Task complete:** A conclusion or summary point in the lab.

**Start lab**

1. To launch the lab, at the top of the page, choose **Start lab**.

 You must wait for the provisioned AWS services to be ready before you can continue.

1. To open the lab, choose **Open Console**.

You are automatically signed in to the AWS Management Console in a new web browser tab.

**Do not change the Region unless instructed.**

COMMON SIGN-IN ERRORS

**Error: You must first sign out**



If you see the message, **You must first log out before logging into a different AWS account:**

* Choose the **click here** link.
* Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
* Choose **Open Console** again.

**Error: Choosing Start Lab has no effect**

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

* Add the lab domain name to your pop-up or script blocker’s allow list or turn it off.
* Refresh the page and try again.

SERVICES USED IN THIS LAB

**Amazon Simple Storage Service (Amazon S3)**

Amazon Simple Storage Service (Amazon S3) is safe, secure, highly scalable object storage in the cloud. It is commonly used for backup and storage, application or media hosting, high-traffic website hosting, or software delivery.

To get the most out of Amazon S3, you need to understand a few simple concepts:

* Amazon S3 stores data as objects within buckets. An object is composed of a file and, optionally, any metadata that describes that file.
* To store an object in Amazon S3, you upload it to a bucket. When you upload an object, you can set permissions on the object as well as any metadata.
* Buckets are the containers for objects. You can have one or more buckets. For each bucket, you can:
  + Control access to the bucket, defining who can create, delete, and list objects in the bucket.
  + View access logs for the bucket and its objects.
  + Choose the geographical region where Amazon S3 stores the bucket and its contents.
* You can create folders to group objects within buckets. You can also nest folders (create folders within folders). If you have used the Amazon S3 API or other utilities, you can learn some important aspects about how folders work with other grouping conventions in the [Organizing objects in the Amazon S3 console by using folders](https://docs.aws.amazon.com/AmazonS3/latest/userguide/using-folders.html) documentation.

**Learn more:** Refer to *Amazon Simple Storage Service Documentation* in the **Additional resources** section for more information.

**Publishing data on Amazon S3**

Imagine that you have a data set that you wish to share with the world. How would you share it if you do not have a web server? If the data set is small enough, you could email it to people or make it available with a file sharing service. But if you have a lot of data, you need something more sophisticated.

Fortunately, Amazon S3 provides a very flexible solution to store and retrieve any amount of data, at any time, from anywhere on the web. Amazon S3 acts like a web server for your data, but does not require you to pay for, nor maintain, a web server. When you store data on Amazon S3, you are only charged for data storage and any requests and data transfer.

When data files are available on Amazon S3, they can be accessed via a simple HTTP endpoint, which makes it easy for people to access programmatically. Publishing data on Amazon S3 also makes it easy to analyze using Amazon Web Services’ computational and data analysis products like Amazon EC2 and Amazon Redshift.

The purpose of this lab is to show you one way to use Amazon S3 to make data available for anyone to access via a web browser.

AWS SERVICES NOT USED IN THIS LAB

AWS service capabilities used in this lab are limited to what the lab requires. Expect errors when accessing other services or performing actions beyond those provided in this lab guide.

**Task 1: Create an Amazon S3 bucket**

Every object in Amazon S3 is stored in a bucket.

In this task, you create a bucket in Amazon S3 to store your data and your website.

1. At the top of the AWS Management Console, in the search bar, search for and choose

S3

.

1. Choose **Create bucket** and then configure:

* **Bucket name:** Enter

websiteNUMBER

 in the textbox.

* + Replace **NUMBER** with a random number.
  + Copy the name of your bucket to a text editor.
* **Region:** Select **US East (N. Virginia) us-east-1** from the dropdown menu.

1. In the **Object Ownership** section configure:

* **ACLs enabled**
* **Object writer**

1. Choose **Create bucket**.

**Note:** Every Amazon S3 bucket must have a unique name.

**Caution:** If you receive an error stating *The requested bucket name is not available*, select the top **Edit** link, change the bucket name, and try again until it works.

After Amazon S3 creates your bucket, the console displays your bucket name in the list of buckets.

**Task complete:** You have successfully created an Amazon S3 bucket.

**Task 2: Upload files**

You can now upload files to your bucket. In this task, you upload a set of sample files.

1. Download the following file: [sample-files.zip](https://us-west-2-tcprod.s3.us-west-2.amazonaws.com/courses/spl-60/v2.2.10.prod-d961811d/scripts/sample-files.zip)
2. Locate the **sample-files.zip** file you downloaded, and then unzip it. It should create a folder with several files.
3. In the **Amazon S3** console, choose the link to the **websiteNUMBER** bucket you created in the previous task.
4. Choose **Upload**.
5. Choose **Add files**.

* A file selection dialog box appears.

1. Browse to the **sample-files** folder that you created from the zip file.
2. Select all of the files, and then choose **Open** (for Windows) or **Open** (for Mac).
3. Choose **Upload**.

The status bar at the bottom of the page displays the upload progress. Verify the files upload successfully.

**Task complete:** You have successfully uploaded a set of sample files to the Amazon S3 bucket.

**Task 3: Grant access to the bucket**

Your files have now been uploaded as objects in your Amazon S3 bucket. The term object is used because files are stored in a filesystem but Amazon S3 is actually an object storage system. While it behaves similar to a filesystem on a computer, it actually has many more capabilities.

Objects in Amazon S3 are private by default. In this task, you verify that the objects in your bucket are not publicly accessible by default.

Information about the object appears on the screen, including:

* **Etag:** An MD5 checksum that can be used to confirm that the object was uploaded correctly.
* **Storage class:** Indicates durability and cost of storing the object.
* **Server side encryption:** Whether the object is encrypted while stored on Amazon S3.
* **Size:** Size of the object being stored.
* **Object URL:** A link to the object on the Internet. Notice that it includes the name of your bucket.

1. Copy the **Object URL** to your clipboard.
2. Open a new browser tab, paste the Object URL link and press Enter.

**Expected output:** You should receive an **Access Denied** message. This is good! It indicates that your data remains private in Amazon S3.

However, when deploying a website, you want your data to be fully accessible on the Internet. You can accomplish this by creating a Bucket Policy that defines who can access specific parts of your bucket. Bucket policies can restrict access only to specific directories, from specific IP addresses, during specific times, and even only to specific users.

1. Keep the **Access Denied** browser tab open. You use it again soon.
2. Switch back to the browser tab open to the **websiteNUMBER - S3 bucket**.
3. At the top of the page, choose **Buckets** to return to the list of buckets.
4. Choose the link to **websiteNUMBER** bucket to view its details.
5. Choose the **Permissions** tab.
6. Navigate to the **Block public access (bucket settings)** section and choose **Edit**.
7. **Deselect** the **Block *all* public access** option, and then leave all other options **deselected**.

**Security:** Notice all of the individual options remain deselected. When deselecting all public access, you must then select the individual options that apply to your situation and security objectives. In a production environment, it is recommended to use the least permissive settings possible.

1. Choose **Save changes**.
2. A dialogue box opens asking you to confirm your changes. Enter

confirm

 in the textbox, and choose **Confirm**.

1. Navigate to the **Bucket policy** section and choose **Edit**.

The **Bucket policy editor** panel opens with a field where you can enter a policy for your bucket.

1. **Copy edit:** Copy and paste the following policy into the **Bucket policy editor** (but don’t save yet):

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadGetObject",

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::BUCKET/\*"

}

]

}

1. In the policy, replace **BUCKET** with the name of your bucket. Keep the

/\*

 in place after your bucket name!

* The \* is a wildcard, so adding

/\*

 at the end of the bucket name applies this policy to all objects in the bucket.

* + You could limit the policy to a specific folder, rather than making the entire bucket public, by adding the path to the folder. For example: “arn:aws:s3:::BUCKET/FOLDER/\*”
* The policy *Allows* anyone (indicated by Principal: \*) to download an object (**GetObject**) from your bucket.

1. Choose **Save changes** to save the new bucket policy.

**Caution:** If you receive an error when saving, it might be due to a mismatch with your bucket name. Make sure the bucket name in the **Resource** line matches the name of the bucket your created earlier (shown at the top of the window).

You can now test that the objects are accessible.

1. Return to the browser tab that previously showed **Access Denied**.
2. **Refresh:** Refresh the web page.

The AWS logo should appear, proving that your Bucket Policy now grants public access to your Amazon S3 bucket.

1. Close the browser tab with the logo.
2. Return to the browser tab open to the **websiteNUMBER - S3 bucket** page.

**Task complete:** You have successfully verified that objects in your bucket are not publicly accessible by default, and granted access to the objects as per your website requirements.

**Task 4: Configure static website hosting**

Amazon S3 also has the ability to configure a bucket for static website hosting, which allows you to:

* Define a **default page** that should be displayed when users access your website but do not specify a particular HTML page.
* Define an **error page** that appears if users navigate to a page that does not exist.
* Define **redirections** to send users to alternate pages at different locations.

In this task, you activate static website hosting and configure it to use the **index.html** and **error.html** pages that were included in your upload.

1. Choose the **Properties** tab at the top of the page.
2. Navigate to the **Static website hosting** section and choose **Edit**.

On the **Edit static website hosting** page, select  **Enable** and configure the following:

* **Index document:** Enter

index.html

 in the textbox. (You must enter these values even though they are already displayed as hints.)

* **Error document:**

error.html

 in the textbox.

1. Choose **Save changes**.
2. Navigate to the **Static website hosting** section again and copy the **Bucket website endpoint** link.

**Note:** This is a link to your new website. It should look similar to:

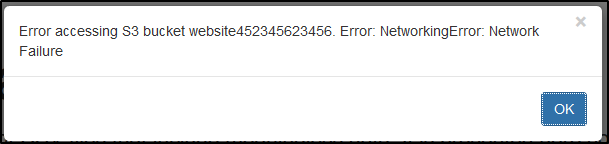
http://website-XXX.s3-website-us-west-2.amazonaws.com

1. Open a new browser tab, paste the **Bucket website endpoint** link, and press enter.

The **index.html** page opens in a new tab.

**Caution:** You also see an error message on the page. This is okay! You still have some configuration to perform.

**Expected output:**



1. Within the error message window, choose **OK**.

Next, you test the **error.html** page.

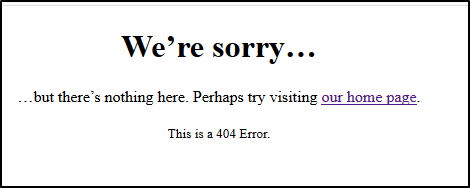
1. In the address bar of your browser, type extra characters at the end of the previous URL to go to a page that does **not** exist.

For example:

http://website-XXX.s3-website-us-west-2.amazonaws.com/wrong

**Note:** The **error.html** page should display with a message that says **We’re sorry…**. This error page is also known as a **404 page** because error 404 means *Page Not Found*.

**Expected output:**



You now have a few final steps to configure your website!

**Task complete:** You have successfully configured static website hosting on your S3 bucket.

**Task 5: Enable Cross-Origin Resource Sharing**

You can also access your Amazon S3 bucket programmatically. This allows the objects in your bucket to be shown within web pages without users even knowing that they are coming from Amazon S3.

To demonstrate this concept, the *index.html* page that you uploaded uses JavaScript to dynamically read the contents of your bucket and display a list of objects to your visitors. To do this, you must enable **Cross-Origin Resource Sharing (CORS)**. A browser would normally block JavaScript from allowing requests to read the contents of your bucket, but with CORS you can configure the bucket to explicitly permit JavaScript to do this.

In this task, you define a CORS configuration to allow cross-origin requests for the bucket. A CORS configuration is an XML document with rules that identify the origins that are permitted to access the bucket, the operations (HTTP methods) it supports for each origin, and other operation-specific information.

**Note:** Cross-Origin Resource Sharing (CORS) defines a way for client web applications that are loaded in one domain to interact with resources in a different domain. With CORS support, you can build rich client-side web applications with Amazon S3 and selectively allow cross-origin access to your Amazon S3 resources.

1. Keep the **We’re Sorry…** browser tab open. You use it again soon.
2. Switch back to the browser tab open to the **websiteNUMBER - S3 bucket** page.
3. Scroll up to the top of the page and choose the **Permissions** tab.
4. Navigation to the **Cross-origin resource sharing (CORS)** section, and choose **Edit**.
5. **Copy edit:** Copy the following policy, and then paste it into the editor:

[

{

"AllowedHeaders": [

"\*"

],

"AllowedMethods": [

"GET"

],

"AllowedOrigins": [

"\*"

],

"MaxAgeSeconds": 3000

}

]

**Security:** This is a basic and very permissive CORS configuration, but Amazon S3 supports a wide array of configuration options. It is recommended to always use the least permissive policies possible.

1. Choose **Save changes**.

There is one final step required, which is to permit programmatic listing of your bucket.

1. Navigate to the **Access control list (ACL)** section, and choose **Edit**.
2. For **Everyone (public access)**, select  **List**.
3. Navigate to the bottom of the page and select  **I understand the effects of these changes on my objects and buckets**.
4. Choose **Save changes**.

**Caution:** A warning appears in the **Access control list (ACL)** section stating: **AWS doesn’t recommend granting access to the Everyone grantee.** This is expected because you are intentionally adding public access.

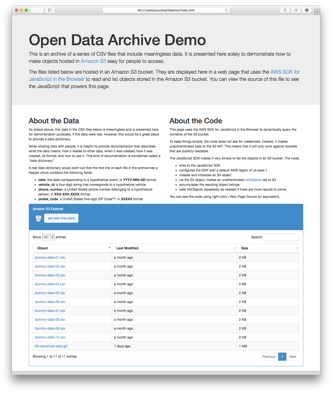
You can now test your application!

1. Return to the **Error - 404 Not Found** tab in your web browser and choose the ***try visiting our home page*** link to return to the application.

**Note:** If you can’t find the tab, choose the **Bucket website endpoint** URL shown in the **Static website hosting** section of the **Properties** tab.

**Expected output:**

The site looks something like this:



**Note:** The page includes documentation for the application and the bottom of the page lists the objects stored in your Amazon S3 bucket.

**Task complete:** You have successfully enabled cross-origin resource sharing (CORS) for your S3 bucket.

**Understand the website**

Feel free to explore the source code of the **index.html** file. It is a static HTML file, but it is made dynamic through the use of several JavaScript plugins and custom JavaScript that displays the contents of an Amazon S3 bucket in an HTML table. The JavaScript includes comments to help you understand the application.

While this lab doesn’t require you to know how to write JavaScript, HTML, or CSS, listed below are a few interesting aspects of the index.html file.

THE AWS SDK FOR JAVASCRIPT

The index.html page uses the AWS SDK for JavaScript in the browser to dynamically query the contents of an S3 bucket.

The JavaScript SDK makes it very simple to list the objects in an S3 bucket. The code:

* Links to the AWS JavaScript SDK.
* Configures the SDK with a default AWS region of us-west-2.
* Creates and initializes an Amazon S3 object.
* Via the Amazon S3 object, makes an unauthenticated *listObjects* call to Amazon S3.
* Accumulates the resulting object listings.
* Calls *listObjects* repeatedly as needed if there are more results to come.

To keep things simple, the code does not ask for credentials when querying a bucket. Instead, it makes unauthenticated calls to the Amazon S3 API. This means that it only works against buckets that are publicly readable. This is why you to set the permissions of your bucket to allow anyone to list the contents of the bucket.

The last point on the list above is worth noting because *listObjects* only returns 1,000 objects at a time. Due to this, the JavaScript code continually queries a bucket to see if there are more objects to retrieve. The code has been tested on buckets with over 10,000 objects and encountered no issues, but it may overwhelm your browser if you try to display the contents of a bucket with a very large amount of objects (say, over one million objects).

The JavaScript only queries for objects at one level of the bucket at a time. That is, it only looks for objects in the root of your bucket and not look for objects in folders unless you choose them.

OTHER FEATURES

The index.html page relies on these other libraries to work:

* **JQuery**, which is a common JavaScript library used to create browser-based apps: http://jquery.com
* **Bootstrap**, for CSS styling: http://getbootstrap.com
* **Font Awesome**, for vector icons: http://fontawesome.io
* **DataTables**, for table organization, filtering, and sorting: http://datatables.net
* **Bootbox**, for popup dialogs and alerts that work with Bootstrap: http://bootboxjs.com
* **Moment**, for time formatting (e.g. “2 days ago”, “last year”): http://momentjs.com

**Conclusion**

You have successfully done the following:

* Created an Amazon S3 bucket.
* Uploaded objects to Amazon S3.
* Set permissions on your bucket by creating a bucket policy.
* Created a CORS configuration policy for the bucket.
* Configured a bucket for static website hosting.
* Used JavaScript to display the contents of a bucket on a static webpage.

**End lab**

Follow these steps to close the console and end your lab.

1. Return to the **AWS Management Console**.
2. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
3. Choose **End lab** and then confirm that you want to end your lab.

**Additional resources**

* [Amazon S3](https://aws.amazon.com/s3/)
* [Amazon Simple Storage Service Documentation](https://aws.amazon.com/documentation/s3/)
* [Hosting a static website using Amazon S3](https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteHosting.html)
* [Tutorial: Configuring a static website using a custom domain registered with Route 53](https://docs.aws.amazon.com/AmazonS3/latest/userguide/website-hosting-custom-domain-walkthrough.html)
* [What Is the AWS SDK for JavaScript?](https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/welcome.html)
* [Using cross-origin resource sharing (CORS)](https://docs.aws.amazon.com/AmazonS3/latest/userguide/cors.html)

For more information about AWS Training and Certification, see [*https://aws.amazon.com/training/*](https://aws.amazon.com/training/).

*Your feedback is welcome and appreciated.*  
If you would like to share any feedback, suggestions, or corrections, please provide the details in our [*AWS Training and Certification Contact Form*](https://support.aws.amazon.com/#/contacts/aws-training).