



AWS Software Development Kit (SDK) for .NET

Author: Luis Coco Enríquez

Agenda

Topic

What is AWS SDK for .NET?.

Most common use cases.

How to set up your environment.

API Reference.

Agenda

Topic

Code examples with guidance for the AWS SDK for .NET.

Additional code examples for the AWS SDK for .NET.

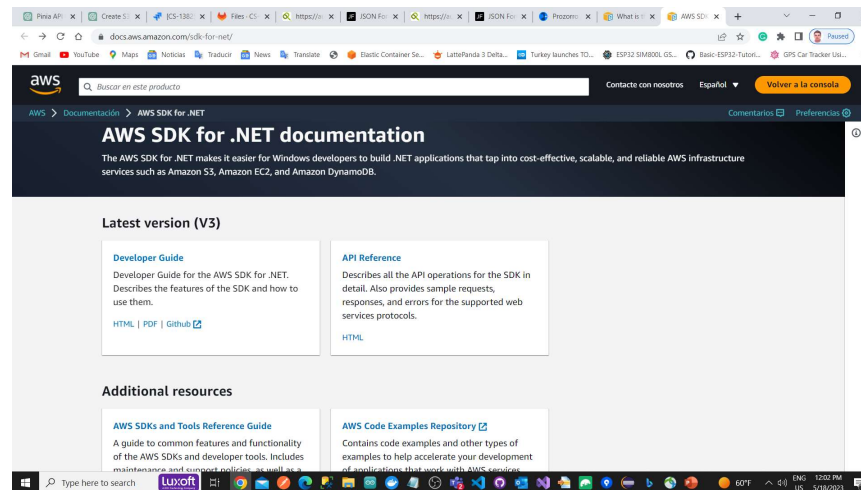
GitHub repositories.

C# code samples with AWS SDK for .NET.

What is AWS SDK for .NET?

The **AWS SDK for .NET** is a set of libraries (APIs) and tools that allow .NET developers to **interact with** various **AWS services using the .NET framework**. It provides a convenient way to access AWS services such as Amazon S3, Amazon DynamoDB, Amazon EC2, and many more.

To use the AWS SDK for .NET, you need to install the **AWSSDK** packages using **NuGet**, which is the core library that provides the foundational functionality for working with AWS services. Additionally, you can install other specific service packages like **AWSSDK.S3** or **AWSSDK.DynamoDB** depending on the services you want to interact with.



<https://docs.aws.amazon.com/sdk-for-net/>

What is AWS SDK for .NET?

The screenshot shows a web browser displaying the AWS SDK for .NET Developer Guide. The browser's address bar shows the URL <https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/welcome.html>. The page has a dark blue header with the AWS logo and a search bar. Below the header, the breadcrumb navigation reads "AWS > Documentation > AWS SDK for .NET > Developer Guide". The main content area is titled "What is the AWS SDK for .NET" and includes a sub-header "PDF | RSS". The text explains that the AWS SDK for .NET simplifies building .NET applications that use AWS services. A diagram illustrates the workflow: a .NET application uses the AWS SDK for .NET to interact with various AWS services, including Amazon S3, Amazon EC2, Amazon IAM, Amazon SQS, Amazon Fargate, Amazon CloudWatch, Amazon DynamoDB, AWS Lambda, and Amazon EKS. The diagram shows a .NET icon connected to the AWS SDK for .NET icon, which then connects to a grid of AWS service icons. The right sidebar, titled "En esta página", lists sections like "About this version", "Maintenance and support for SDK major versions", "Common use cases", and "Additional topics in this section". The bottom of the page features a Windows taskbar with various application icons and a system tray showing the date and time as 12:00 PM on 5/18/2023.

What is the AWS SDK for .NET

The AWS SDK for .NET makes it easier to build .NET applications that tap into cost-effective, scalable, and reliable AWS services such as Amazon Simple Storage Service (Amazon S3) and Amazon Elastic Compute Cloud (Amazon EC2). The SDK simplifies the use of AWS services by providing a set of libraries that are consistent and familiar for .NET developers.

(OK, got it! I'm ready for a [quick tour](#) or to start [setting up](#).)

<https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/welcome.html>

Most common use cases

The AWS SDK for .NET provides a wide range of functionalities that enable developers to build various types of applications and services using the .NET framework. Some of the most common use cases for the AWS SDK for .NET are:

Building web applications: You can leverage the SDK to interact with various AWS services like Amazon S3 for storage, Amazon DynamoDB for NoSQL database, Amazon SES for sending emails, Amazon SQS for message queuing, and many more.

Serverless computing: The AWS SDK for .NET integrates with AWS Lambda, allowing you to build serverless applications using .NET. You can create Lambda functions, interact with other AWS services, and respond to events using the SDK.

Data processing and analytics: The SDK provides APIs to work with AWS services like Amazon Redshift for data warehousing, Amazon EMR for big data processing, Amazon Kinesis for real-time streaming data, and Amazon Athena for interactive query analysis.

Internet of Things (IoT): You can use the SDK to develop IoT applications by interacting with AWS IoT Core, which enables secure communication between devices and the cloud.

Most common use cases

Machine Learning: The AWS SDK for .NET supports AWS services like Amazon SageMaker, Amazon Rekognition, and Amazon Comprehend, allowing you to build and deploy machine learning models and perform tasks like image recognition, natural language processing, and more.

DevOps and Infrastructure Automation: The SDK integrates with AWS CloudFormation for infrastructure as code, AWS Systems Manager for managing resources, AWS CodeDeploy for deploying applications, AWS CloudWatch for monitoring and logging, and other services to automate and manage your infrastructure and deployments.

Mobile and gaming applications: The SDK provides support for AWS Mobile services, including Amazon Cognito for user authentication and authorization, Amazon Mobile Analytics for tracking app usage, Amazon Pinpoint for user engagement, and Amazon GameLift for building and scaling multiplayer games.

Integration with third-party libraries and frameworks: The AWS SDK for .NET is compatible with popular .NET libraries and frameworks, allowing you to integrate AWS services seamlessly into your existing applications or frameworks like ASP.NET, Xamarin, Unity, and more.

How to setup your environment

Follow these steps to install and run the AWS SDK for .NET:

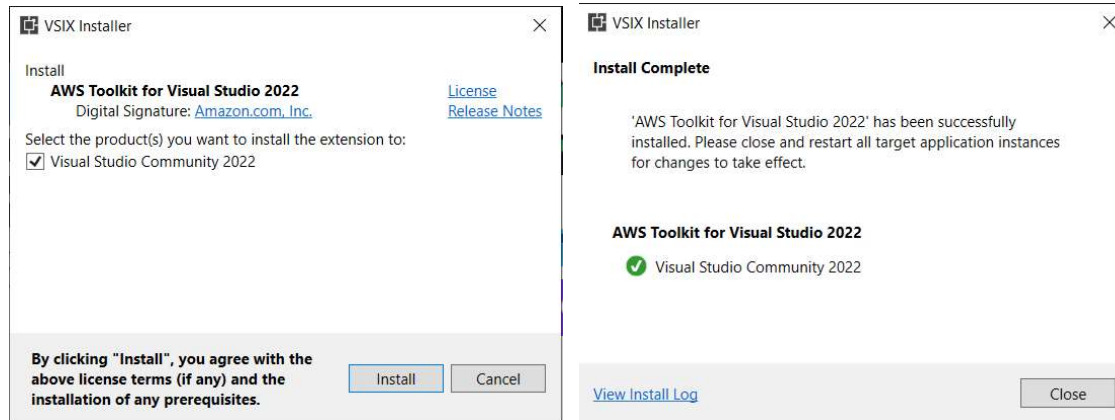
1. Create a free AWS account. See video: [How to sign up for the AWS free tier in 2023](#)
2. Create an administrative user. See video: [How to create Admin User for AWS account](#)
3. Login with the new user and See video: [How to get AWS access key and secret key](#)
4. Install AWS CLI. See video: [How to download install and configure AWS CLI on Windows](#)
5. Run the command “aws configure” to configure your AWS account (enter your region “eu-west-3”, your default output format “json”). See video: [AWS CLI Tutorial](#)
6. (Optional) You can create different profiles and switch between them (see the following slides).

How to create and modify AWS profiles

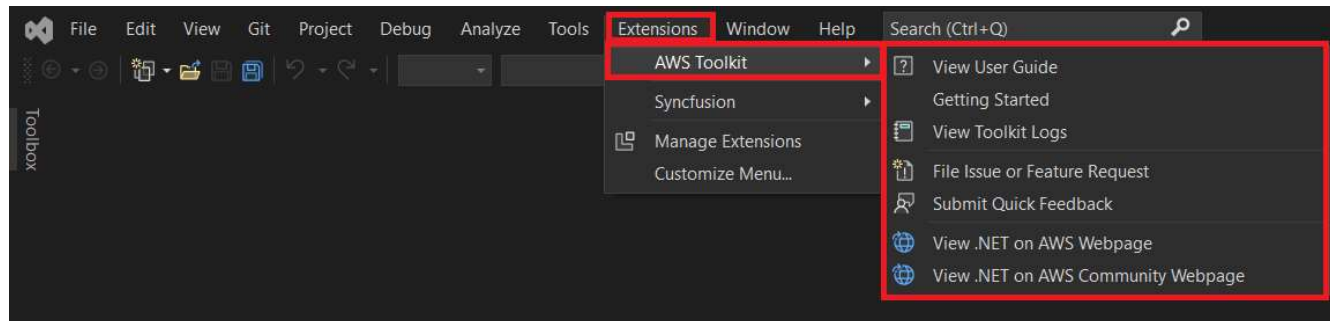
1. Open a command line as administrator.
2. For **listing** your current **profile**: **aws configure list**
For **listing all** your AWS **profiles** run the command: **aws configure list-profiles**
3. To **add a new profile**, first login in AWS Console and generate an `access_key_id` and a `secret_access_key`.
Then run the command: **aws configure --profile second_user**.
Enter the profile region (for example: eu-west-3) and default output format (for example: json)
4. Profiles are stored under the “**config**” and “**credentials**” files: “C:\Users\LEnriquez\.aws”
5. To switch between different profiles run the command: **set AWS_PROFILE=profile_name**
<https://www.simplified.guide/aws/cli/configure-multiple-profiles>
<https://dev.to/hmintoh/how-to-use-multiple-aws-accounts-with-the-aws-cli-3lge#:~:text=To%20switch%20between%20different%20AWS,variable%20to%20a%20different%20value.>

(Optional) Install AWS Toolkit for Visual Studio 2022

Download and install AWS Toolkit <https://aws.amazon.com/es/visualstudio/>

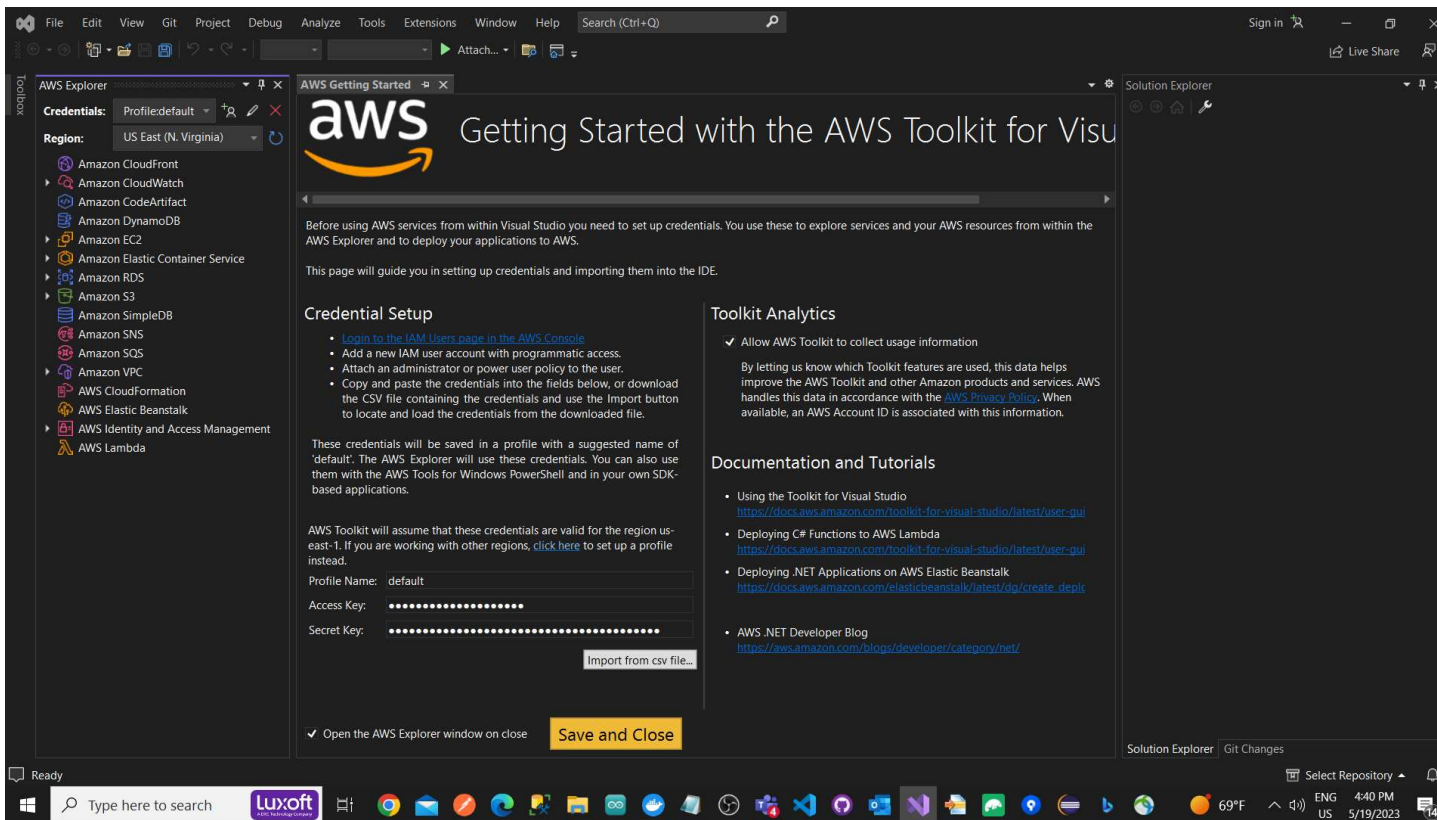


Run Visual Studio and select the menu option: Menu->**AWS Toolkit->Getting Started**



(optional) Install AWS Toolkit for Visual Studio 2022

Select in the Menu->View>**AWS Explorer**



API reference

The screenshot shows a web browser window displaying the AWS SDK for .NET Version 3 API Reference page. The browser's address bar shows the URL `docs.aws.amazon.com/sdkfornet/v3/apidocs/index.html`. The page has a sidebar on the left with a list of links to various AWS services, including Amazon, Amazon AccessAnalyzer, Amazon AccessAnalyzer.Model, Amazon Account, Amazon Account.Endpoints, Amazon Account.Model, Amazon ACMPCA, Amazon ACMPCA.Endpoints, Amazon ACMPCA.Model, Amazon AlexaForBusiness, Amazon AlexaForBusiness.Endpoint, Amazon AlexaForBusiness.Model, Amazon Amplify, Amazon Amplify.Endpoints, Amazon Amplify.Model, Amazon Amplify.Backend, Amazon Amplify.Backend.Endpoint, Amazon Amplify.Backend.Model, Amazon Amplify.UIBuilder, Amazon Amplify.UIBuilder.Endpoint, Amazon Amplify.UIBuilder.Model, Amazon APIGateway, Amazon APIGateway.Endpoints, Amazon APIGateway.Model, Amazon ApiGatewayManagement, Amazon ApiGatewayManagement/, Amazon ApiGatewayManagement/, Amazon ApiGatewayManagement/, and Amazon ApiGatewayV2. The main content area is titled "AWS SDK for .NET Version 3 API Reference" and contains a search bar, a paragraph about the SDK, and a section titled "Additional Information" with links to "Getting Started with the AWS SDK for .NET", "Setting up a project", "Programming AWS", "AWS Developer Blog - Windows and .NET", "V3 Migration Guide", "V3.5 Migration Guide", and "Migrating from .NET Standard 1.3". The Windows taskbar is visible at the bottom, showing the search bar, the Luxoft logo, and various application icons.

<https://docs.aws.amazon.com/sdkfornet/v3/apidocs/index.html>

API reference

The screenshot shows a web browser window displaying the AWS SDK for .NET API documentation. The address bar shows the URL: <https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/S3/MS3PutBucketAsyncStringCancellationToken.html>. The page title is "AmazonS3Client.PutBucketAsync (string, CancellationToken)". The left sidebar contains a navigation menu with links to various API classes and methods, including AmazonS3Client, AmazonS3Config, AmazonS3DefaultConfiguration, AmazonS3Exception, AnalyticsS3ExportFileFormat, ArchiveStatus, ArnExtensions, BucketAccelerateStatus, ChecksumAlgorithm, ChecksumMode, CompressionType, DeleteMarkerReplicationStatus, DeleteObjectsException, EncodingType, EventType, ExistingObjectReplicationStatus, ExpressionType, FileHeaderInfo, GlacierJobTier, GranteeType, HttpVerb, and IAmazonS3. The main content area is titled "METHOD" and contains the following text:

Creates a new S3 bucket. To create a bucket, you must register with Amazon S3 and have a valid Amazon Web Services Access Key ID to authenticate requests. Anonymous requests are never allowed to create buckets. By creating the bucket, you become the bucket owner.

Not every string is an acceptable bucket name. For information about bucket naming restrictions, see [Bucket naming rules](#).

If you want to create an Amazon S3 on Outposts bucket, see [Create Bucket](#).

By default, the bucket is created in the US East (N. Virginia) Region. You can optionally specify a Region in the request body. You might choose a Region to optimize latency, minimize costs, or address regulatory requirements. For example, if you reside in Europe, you will probably find it advantageous to create buckets in the Europe (Ireland) Region. For more information, see [Accessing a bucket](#).

If you send your create bucket request to the `s3.amazonaws.com` endpoint, the request goes to the us-east-1 Region. Accordingly, the signature calculations in Signature Version 4 must use us-east-1 as the Region, even if the location constraint in the request specifies another Region where the bucket is to be created. If you create a bucket in a Region other than US East (N. Virginia), your application must be able to handle 307 redirect. For more information, see [Virtual hosting of buckets](#).

Access control lists (ACLs)

When creating a bucket using this operation, you can optionally configure the bucket ACL to specify the accounts or groups that should be granted specific permissions on the bucket.

If your CreateBucket request sets bucket owner enforced for S3 Object Ownership and specifies a bucket ACL that provides access to an external Amazon Web Services account, your request fails with a 400 error and returns the `InvalidBucketACLWithObjectOwnership` error code. For more information, see [Controlling object ownership](#) in the *Amazon S3 User Guide*.

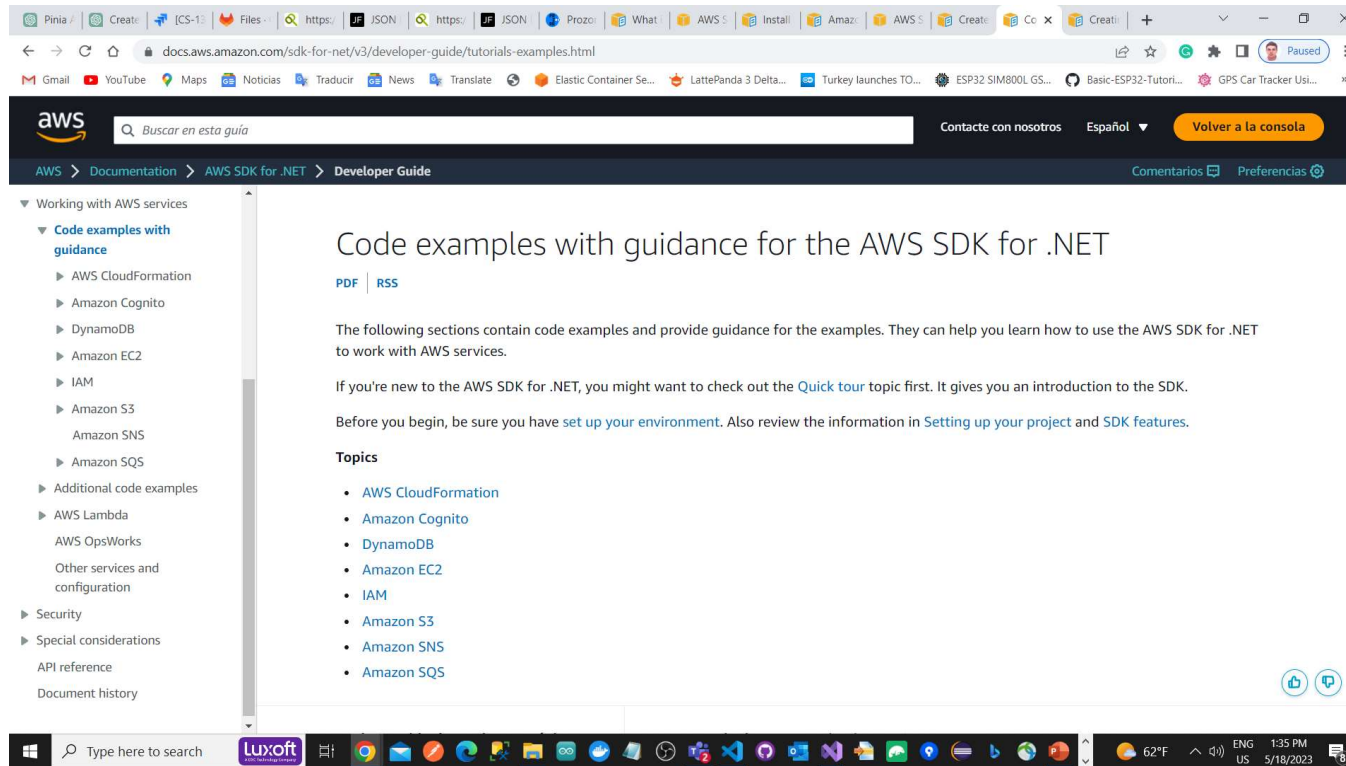
There are two ways to grant the appropriate permissions using the request headers.

- Specify a canned ACL using the `x-amz-acl` request header. Amazon S3 supports a set of predefined ACLs, known as *canned ACLs*. Each canned ACL has a predefined set of grantees and permissions. For more information, see [Canned ACL](#).
- Specify access permissions explicitly using the `x-amz-grant-read`, `x-amz-grant-write`, `x-amz-grant-read-acp`, `x-amz-grant-write-acp`, and `x-amz-grant-full-control` headers. These headers map to the set of permissions Amazon S3 supports in an ACL. For more information, see [Access control list \(ACL\) overview](#).

You specify each grantee as a `type=value` pair, where the type is one of the following:

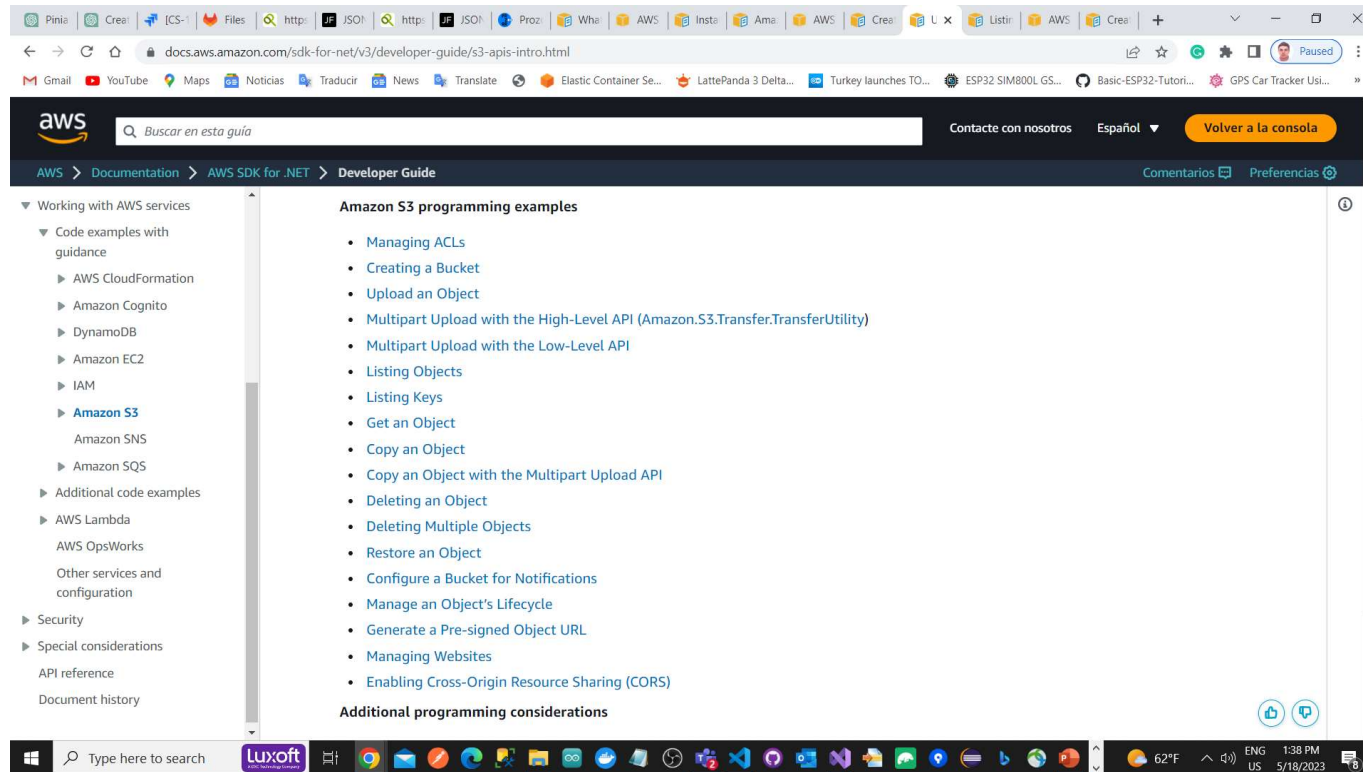
<https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/S3/MS3PutBucketAsyncStringCancellationToken.html>

Code examples with guidance for the AWS SDK for .NET



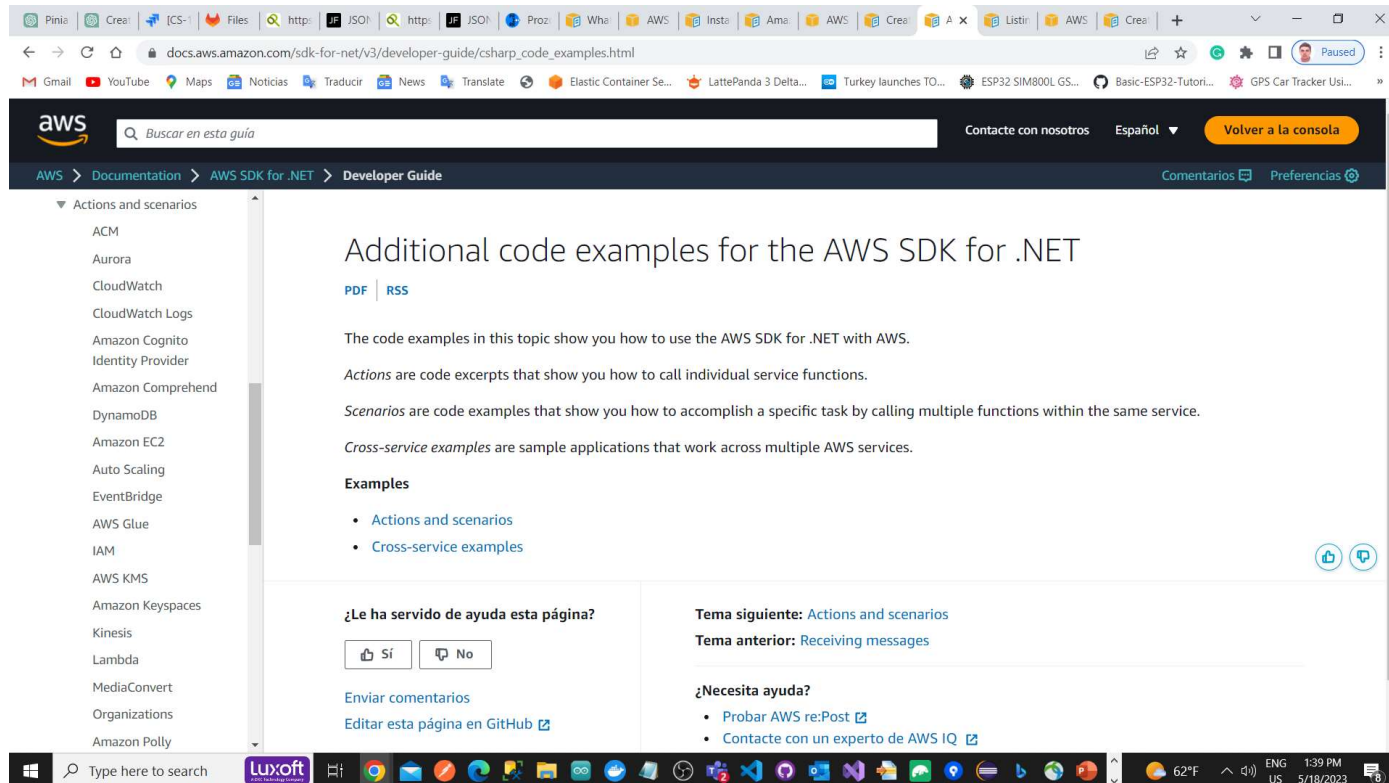
<https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/tutorials-examples.html>

Code examples with guidance for the AWS SDK for .NET



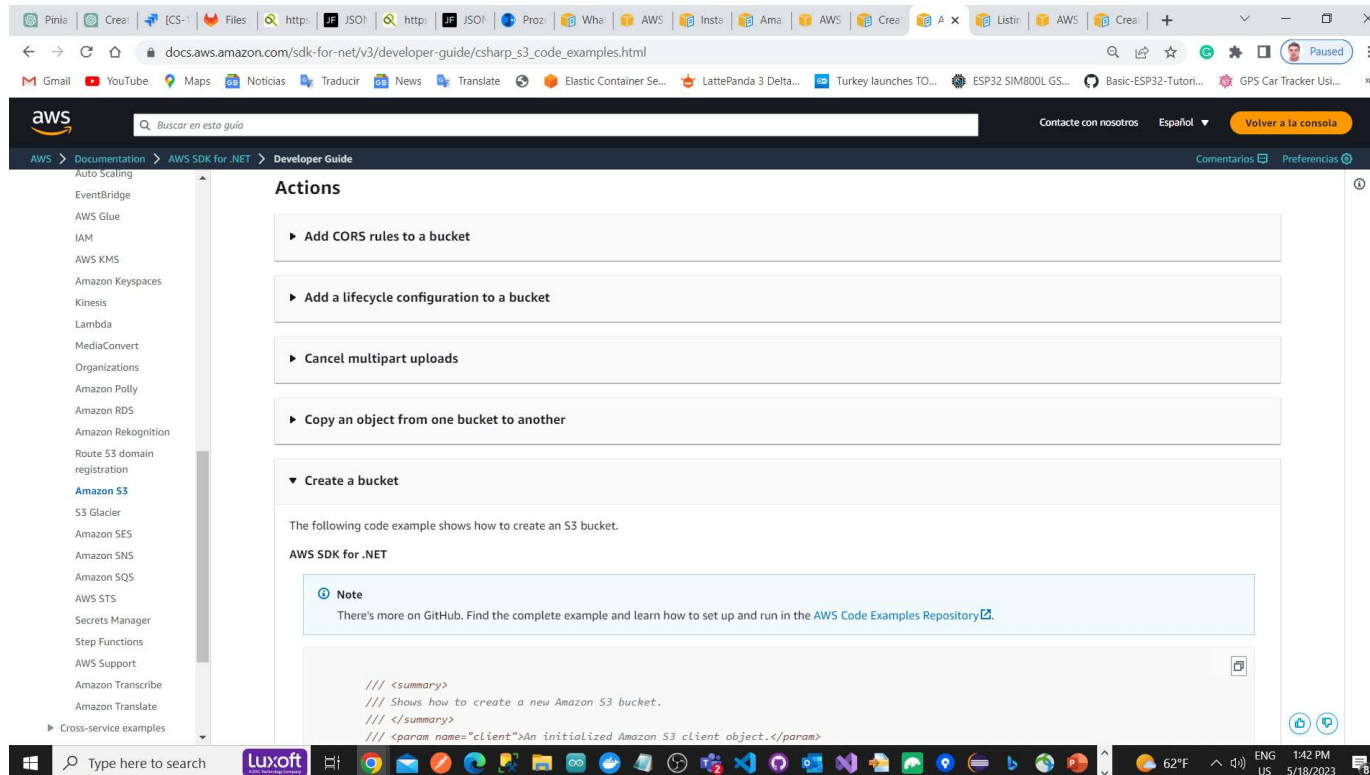
<https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/s3-apis-intro.html>

Additional code examples for the AWS SDK for .NET



https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/csharp_code_examples.html

Additional code examples for the AWS SDK for .NET

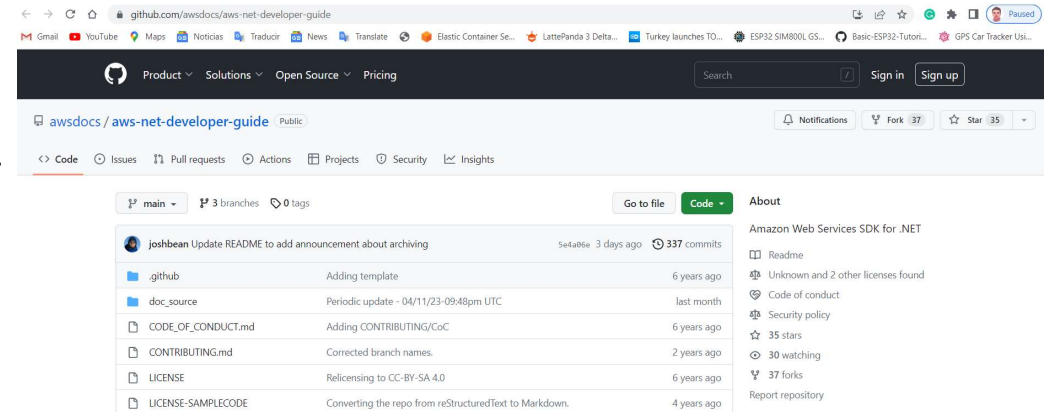


https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/csharp_s3_code_examples.html

GitHub repositories

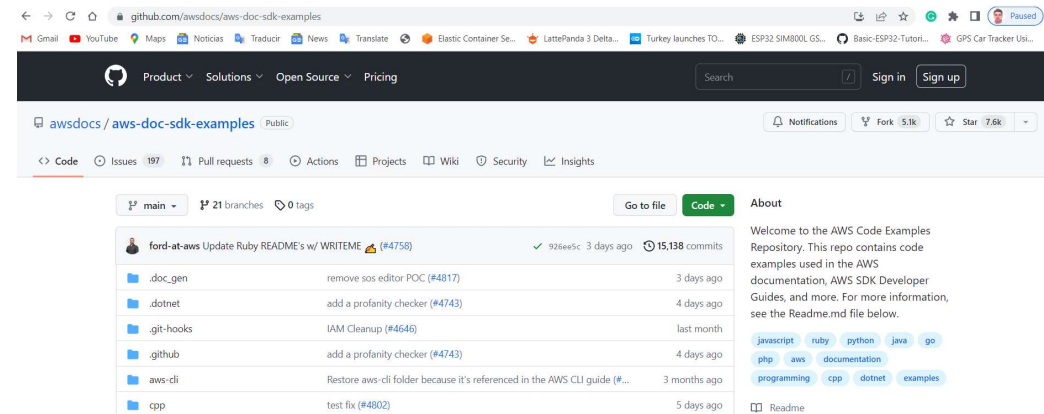
The Developer guide for the AWS SDK for .NET

<https://github.com/awsdocs/aws-net-developer-guide>



AWS SDK for .NET 3.x documentation examples

<https://github.com/awsdocs/aws-doc-sdk-examples/tree/main/dotnetv3>



C# code samples with AWS SDK for .NET

Sample 1: create a S3 bucket

https://docs.aws.amazon.com/AmazonS3/latest/userguide/example_s3_CreateBucket_section.html

Step 1: Create a new Console .NET project

Start by creating a new .NET project in your preferred IDE (e.g., Visual Studio).

Step 2: Add the necessary NuGet package: add the **AWSSDK.S3**

Step 3: Write the code

Open the code file Program.cs. Import the required namespaces at the top of the file:

```
using Amazon.S3;
using Amazon.S3.Model;

IAmazonS3 client = new AmazonS3Client();
var request = new PutBucketRequest
{
    BucketName = "luiscocoenriquezbucket",
    UseClientRegion = true,
};
var response = await client.PutBucketAsync(request);
```

C# code samples with AWS SDK for .NET

Sample 2: list S3 buckets

https://docs.aws.amazon.com/AmazonS3/latest/userguide/example_s3_ListBuckets_section.html

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
...
var credentials = new Amazon.Runtime.BasicAWSCredentials(accessKey, secretKey);
var config = new AmazonS3Config
{
    RegionEndpoint = RegionEndpoint.USSouth1 // Replace with your desired region
};

using (var client = new AmazonS3Client(credentials, config))
{
    var request = new ListBucketsRequest();
    var response = client.ListBuckets(request);

    foreach (var bucket in response.Buckets)
    {
        Console.WriteLine(bucket.BucketName);
    }
}
```

C# code samples with AWS SDK for .NET

Sample 3: upload an object to a S3 bucket

https://docs.aws.amazon.com/AmazonS3/latest/userguide/example_s3_PutObject_section.html

```
...
var putRequest = new PutObjectRequest

{
    BucketName = bucketName,
    Key = keyName,
    ContentBody = "sample text",
    ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256,
};

var putResponse = await client.PutObjectAsync(putRequest);

// Determine the encryption state of an object.
GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest
{
    BucketName = bucketName,
    Key = keyName,
};
GetObjectMetadataResponse response = await client.GetObjectMetadataAsync(metadataRequest);
ServerSideEncryptionMethod objectEncryption = response.ServerSideEncryptionMethod;
Console.WriteLine($"Encryption method used: {0}", objectEncryption.ToString());
...
```

C# code samples with AWS SDK for .NET

Sample 4: EC2 instance actions

Sample 5: Simple Queue Service (SQS)

Sample 6: DynamoDB

Sample 7: IAM Identity and Access Management services

Sample 8: SES (Simple Email Service)

Sample 9: Autoscaling Groups (EC2)

Sample 10: Aurora (RDS)

Sample 11: EventBridge

Sample 12: RDS (Relational Database Service)

Sample 13: KMS (Keys Management System)

Sample 14: Secrets Manager

Sample 15: SNS (Simple Notification Service)

Sample 16: Lambda

All these examples source code can be downloaded from the public GitHub repository:

<https://github.com/luisccoco/>

