1.6 Storage Services

Amazon Simple Storage Service (S3)

- Object storage
- Storage, protection and management

Amazon S3 Storage Classes

- Different use cases
- Access tiers

Amazon S3 Storage Management

- S3 Lifecycle
- S3 Object Lock
- S3 Replication
- S3 Batch Operation

Amazon S3 Access Management

- S3 Block Public Access
- AWS Identity and Access Management (IAM): recommended
- Bucket Policies
- S3 Access Points
- Access Control List (ACLs): granular control
- S3 Object Ownership

- Access Analyzer for S3

Amazon S3 Data Processing

- S3 Object Lambda: filtering, automatic imagery sizing, redactions, and more
- Event notification: most implemented service

Amazon S3 Analytics and Insights

- Amazon S3 Storage Lens
- Storage Class Analysis
- S3 Inventory with Inventory Reports

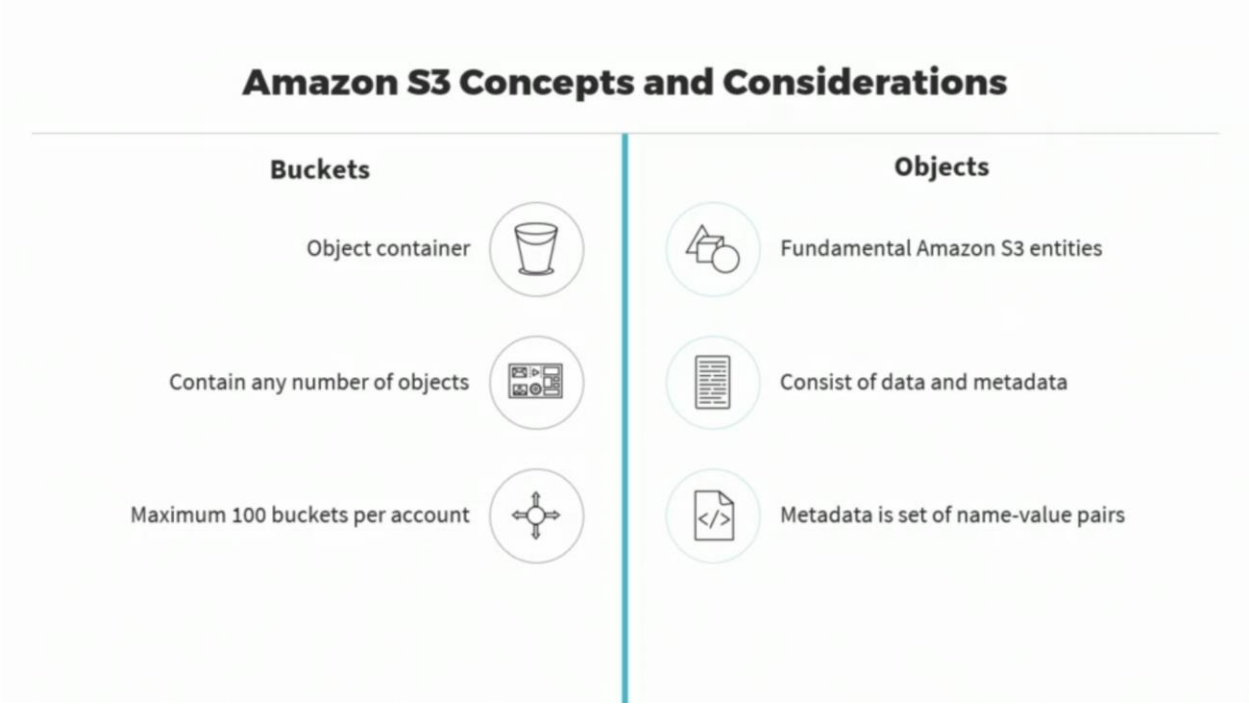
Amazon S3 Strong Consistency

- Read-after-write consistency for PUT and DELETE requests
- Strongly consistent read operations

S3

- Now within S3, there are two main components: buckets and objects.
- buckets contain objects. They are a means of isolation.
- don't think of them as directories on a hard drive. Instead, think of buckets as the hard drive itself. That's how isolated they are from each other.
- objects are the fundamental S3 entities.
- consider the objects as the files and directories inside the bucket.
- Buckets can contain any number of objects, like a limitless hard drive, and objects consist of data and metadata.

- So, if the data is the file content, then the metadata is the information about the file, like its size, type, ACLs and so on.
- So, for buckets you can have a maximum of 100 buckets per AWS account, and object metadata is limited to name-value pairs.
- Keys: A key is the unique identifier for an object within a bucket. A key doesn't have to be globally unique. It just has to be unique within a bucket.



Amazon S3 Concepts and Considerations

- Keys
- Versioning
- Version ID
- Bucket policy
- S3 Access Points
- Access Control Lists (ACLs)
- Regions

Amazon S3 Security

Amazon S3 Data Protection

- Durable infrastructure

- Redundant Storage
- Auto-healing
- Versioning
- IAM
- FIPS endpoint for FIPS 140-2 validation

Amazon S3 Encryption

- Client side
- Server side

Amazon S3 Client-Side Encryption

- Customer Master Key (CMK)
- Option to store key in app
- AWS encryption SDK (Software Development Kit): recommended

Amazon S3 Server-Side Encryption

- Key types
    1. S3 managed keys (SSE-S3): unique key
    2. Server-side encryption with customer master keys (SSE-KMS): customer provided key
    3. Server-side encryption with customer-provided keys (SSE-C): customer provided key

AWS Identity and Access Management (IAM)

- Private by default
- Resource-based policy
- User-based policy

Amazon S3 Access Management

- Who can access
- What resources can be accessed
- Type of actions

Amazon S3 Logging and Monitoring

- Amazon CloudWatch Alarms

- AWS CloudTrail logs
- Amazon S3 access logs
- AWS Trusted advisor

Amazon S3 Development

Amazon S3 Requests

- REST service
- REST API or AWS SDK
- Transactions authenticated or anonymous
- Libraries generate a signature
- REST API calls require code to be written

Amazon S3 Requests and IPv6

- S3 buckets
- IAM policies
- Testing
- Dual-stack endpoints

Amazon S3 dual-stack endpoints

- IPv4 or IPv6
- REST API and endpoint URI
- Accessing S3 buckets
- Region must be specified
- URI be like: s3.dualstack.yourregion.amazonaws.com

Amazon S3 requests and AWS SDK

- AWS account
- IAM credentials
- IAM user temporary credentials
- Federated user temporary credentials

Amazon S3 requests and REST API

- Construct hostnames
- Virtual hosted and path requests
- Request to dual-stack endpoints
- URI: mybucket.s3.dualstack.yourregion.amazonaws.com

- URI: s3.yourregion.amazonaws.com/mybucket (if don’t have dual-stack endpoints)

Request Redirection and REST API

- HTTP user agents
- 100-continue

Amazon S3 Bucket Virtual Hosting

- First slash-delimited part of Request-URI path
- HTTP host header

Amazon S3 Glacier

- Archiving and backup
- Secure and durable
- Long-term storage
- Offload administrative overhead
- Expose RESTful web service
- Called via HTTP/S
- Communicate via JavaScript Object Notation (JSON)
- Request to S3 glacier web service API

Amazon S3 Glacier Archive Storage Classes

- S3 Glacier Instant Retrieval: within seconds
- S3 Glacier Flexible Retrieval: takes up to 5min to 12 hours
- S3 Glacier Deep Archive: can do any time and take 12 hours

Amazon S3 Glacier Objects

- Vault: A vault is a partition within S3 Glacier and inside a vault you would store archives.
- Archive: archives go into vaults, designed for long-term storage. They have an archive ID and not much else. There isn't really directories or metadata or anything like that
- Jobs: A job is any action that you perform on an archive. So, if you wanted to upload or download an archive, you'd have to create a job to do that. And asking for data could take hours, so obviously you'd want to kick off a job and then come back later when it's done.
- Notifications: the signal that you get back when the jobs are completed.

Amazon S3 Glacier Regions and Endpoints

- Vaults are region specific
- Vault requests require specified regions

Creating Vaults

- Maximum 1000 per AWS region
- Naming convention: have unique name within that region

Retrieving Vault Metadata

- API calls
- ASCII values
- Maximum 1000 vaults
- Additional requests

Amazon S3 Glacier Archives

- Unique ID
- Optional description

Uploading Archives

- Single operation up to 4 GB
- Parts total limit of 40 TB

Downloading Archives

- Not available through management console
- Use AWS CLI, REST API or Amazon SDK

Amazon S3 Glacier and Amazon SDKs

- Wrap S3 Glacier API
- AWS security credentials
- Libraries map to REST API and provide objects

Amazon S3 Shared Responsibility

- Security of the cloud
- Security in the cloud

Amazon Elastic Block Store (EBS)

- Block-level volumes for EC2 instances
- Mounted as devices
- Exposed as storage volumes
- Independent persistence
- File system on top of volumes
- Block devices
- Dynamic configuration changes

Working with Amazon EBS Volumes

- Dynamically increase size
- Change IOPS capacity
- Change volume type

Amazon EBS Volume Types

- Solid state drives (SSDs): great performance and considered best volume type for using. Bit costly
- Hard disk drives (HDDs): better for large data, cheaper
- Previous generation (unoptimized): if no upper req., can use this

Amazon EBS Volume Limitations

- Storage capacity
- Service: limit only 64 TB
- Partitioning: 32 bit or 64-bit, limit of 64 zettabyte capacity
- Data block size: bigger the volume, bigger the block

Amazon EBS Snapshots

- Data backup
- Incremental
- Restore from any snapshot

Amazon EBS Data Services

- Amazon EBS Elastic Volumes
- Amazon EBS encryption
- Amazon EBS fast snapshot restore (FSR)

Amazon EBS and NVMe on Linux

- Nitro system

- NVMe device names
- Block device driver

Amazon EBS-optimized Instances

- Optimized configuration stack
- Dedicated amazon EBS I/O capacity
- Dedicated bandwidth
- Superior performance

Amazon Elastic File System (EFS)

- Serverless
- Scale on demand
- Web services interface
- AWS managed infrastructure
- NFS v. 4.1 and 4.0
- Multiple compute instances
- Pay-per-use

Amazon EFS Storage Classes

- Many options
- Standard storage classes
- One zone storage classes

How Amazon EFS Works

- Amazon EC2 and compute instances
- AWS direct connect and AWS managed VPN
- AWS backup
- Authentication and access control
- Data consistency
- Storage classes and lifecycle management

Working with Amazon EFS

- Create file system
- Create mount targets
- Create security groups

Amazon Resource IDs

- Unique resource ID
- Command-line tools and EFS API
- Resource ID length

File Systems and Amazon EFS

- EC2 instance
- NFS permissions
- Only root user has R-W-X (Read Write Execute) access by default
- Single root directory

Mounting Amazon EFS File Systems

- Numerous options
- EFS mount helper

How AWS Backup Works

- Data protection service
- Fully managed
- Centralization and automation

AWS Backup Features

- Backup policies
- Activity monitoring
- Task automation
- No scripts or intervention required

How AWS Backup Works

- Other AWS services: EC2, EFS, EBS, DynamoDB, S3
- Metering and pricing: EFS has different pricing model

AWS Backup Plan Management

- Create plan
- Assign resources
- Delete a plan
- Update a plan

Backup Vaults

- Backup containers

- AWS Key Management Service (AWS KMS)
- Multiple vaults

Working with Backups

- each backup you create is a snapshot in time of your individual resources.
- Recovery points
- Unique ID
- AWS resource types

AWS Backup Security

- Data protection
- IAM
- Compliance validation
- Resilience
- Infrastructure security

AWS Backup Data Protection

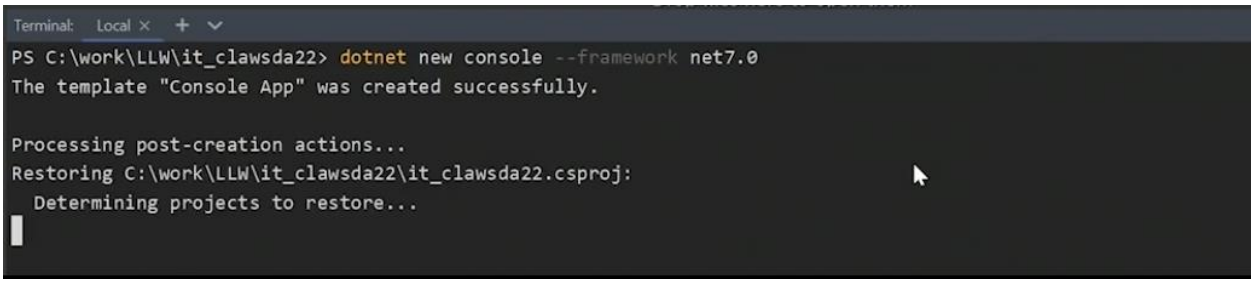
- IAM: no one has access of backup, have to grant explicit permission
- MFA, SSL/TLS, encryption

Backup Encryption

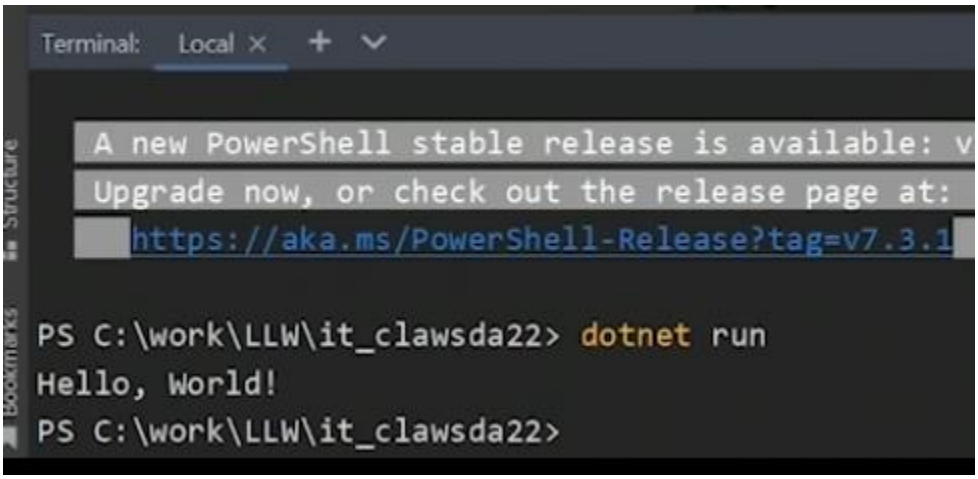
- Default AWS KMS key
- Customer-managed key

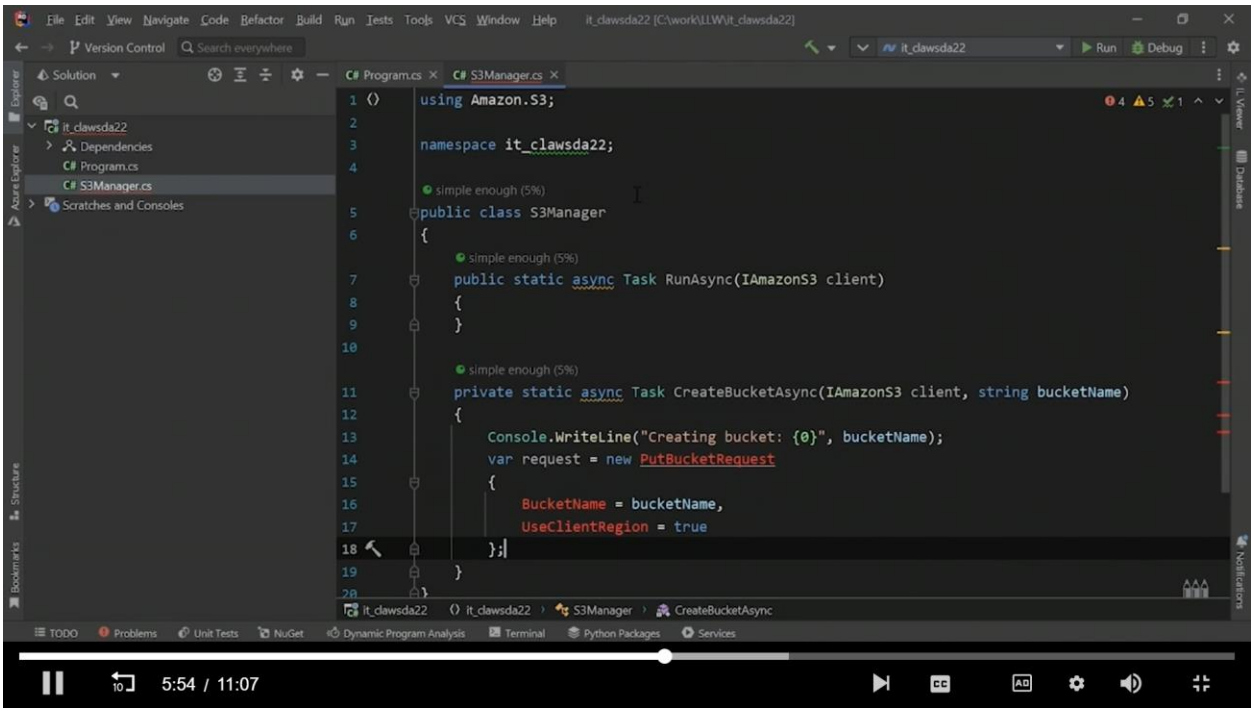
Using Amazon S3

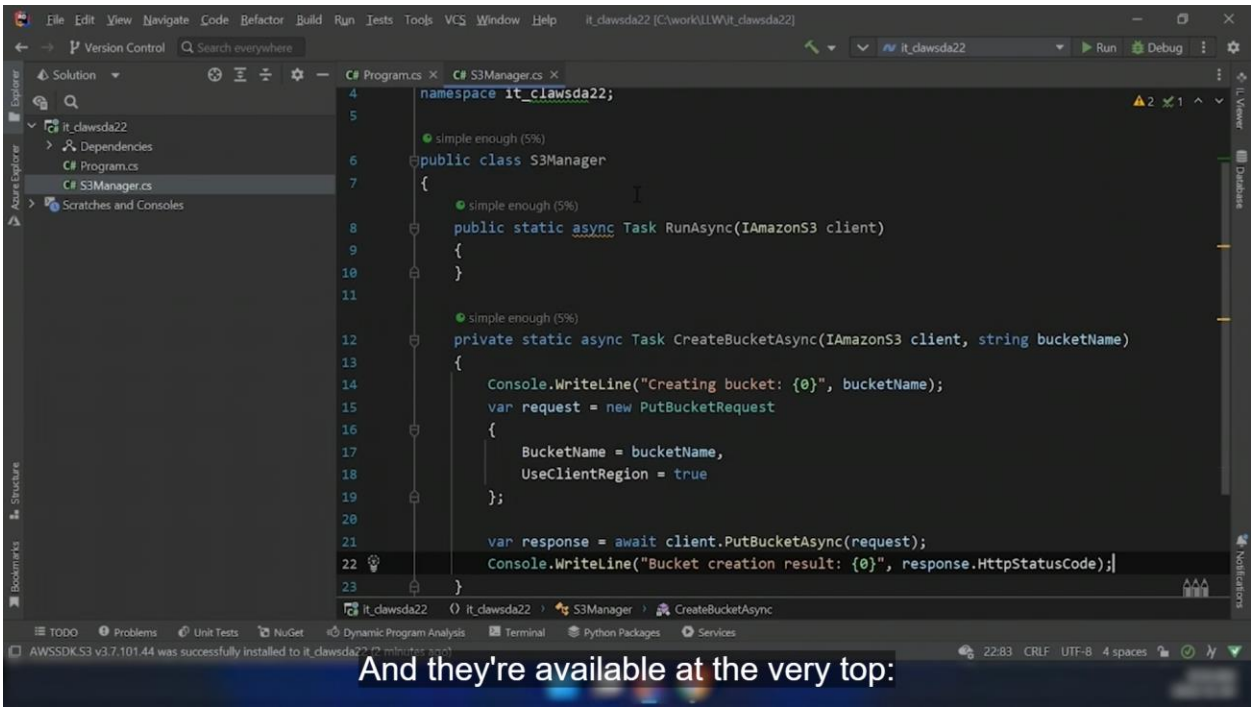
Creating Amazon S3 Bucket

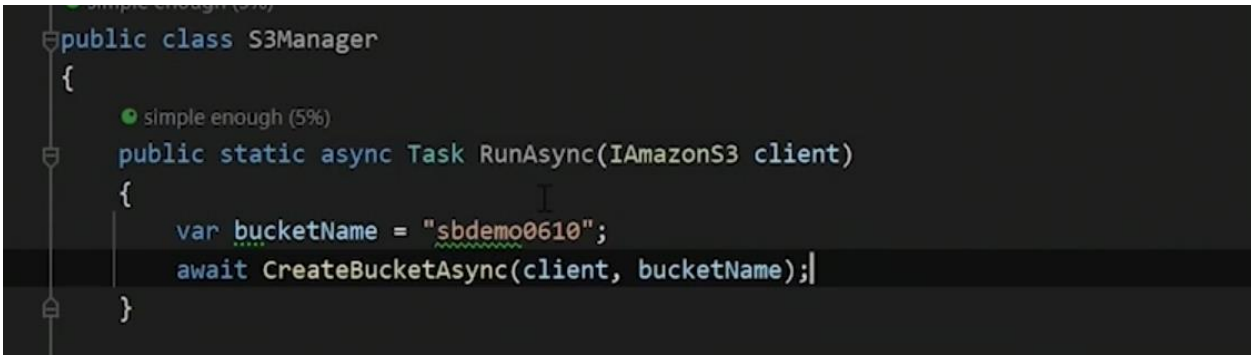


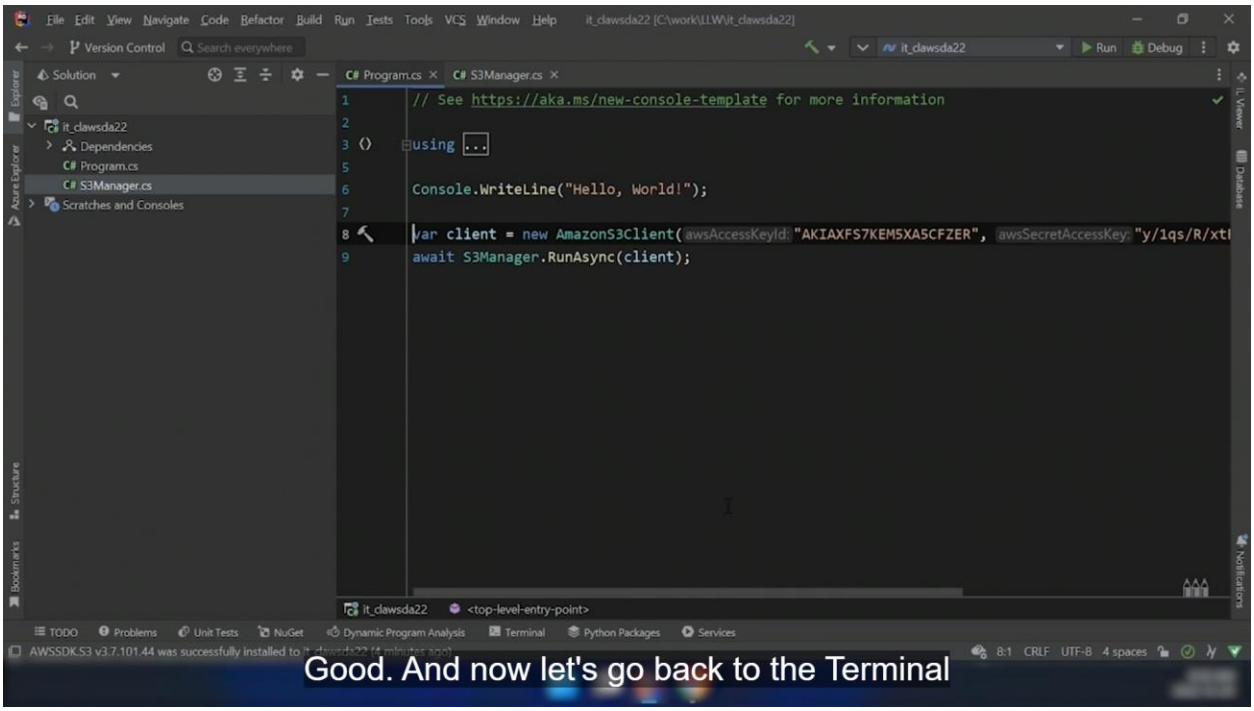
In vs Code terminal

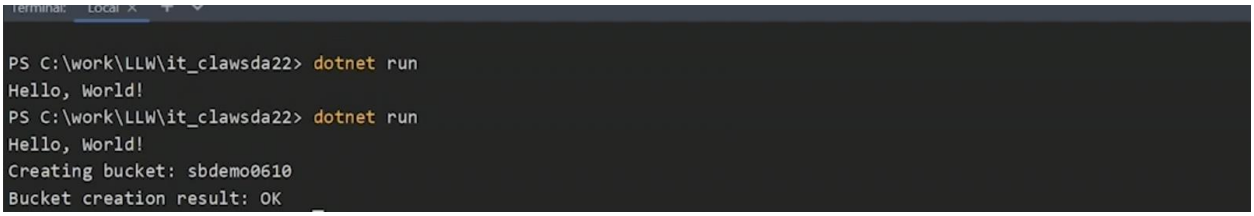

Terminal: Local
A new PowerShell stable release is available: v
Upgrade now, or check out the release page at:
https://aka.ms/PowerShell-Release?tag=v7.3.1
PS C:\work\LLW\it_clawsda22> dotnet run
Hello, World!
PS C:\work\LLW\it_clawsda22>


File Edit View Navigate Code Refactor Build Run Tests Tools VCS Window Help
it_clawsda22 [C:\work\LLW\it_clawsda22]
Version Control
Search everywhere
it_clawsda22
Run
Debug
Solution
Program.cs
S3Manager.cs
it_clawsda22
Dependencies
Program.cs
S3Manager.cs
Scratches and Consoles
using Amazon.S3;
namespace it_clawsda22;
simple enough (5%)
public class S3Manager
{
simple enough (5%)
public static async Task RunAsync(IAmazonS3 client)
{
}
simple enough (5%)
private static async Task CreateBucketAsync(IAmazonS3 client, string bucketName)
{
Console.WriteLine("Creating bucket: {0}", bucketName);
var request = new PutBucketRequest
{
BucketName = bucketName,
UseClientRegion = true
};
}
it_clawsda22 it_clawsda22 S3Manager CreateBucketAsync
TODO
Problems
Unit Tests
NuGet
Dynamic Program Analysis
Terminal
Python Packages
Services
5:54 / 11:07


namespace it_clawsda22;
simple enough (5%)
public class S3Manager
{
simple enough (5%)
public static async Task RunAsync(IAmazonS3 client)
{
}
simple enough (5%)
private static async Task CreateBucketAsync(IAmazonS3 client, string bucketName)
{
Console.WriteLine("Creating bucket: {0}", bucketName);
var request = new PutBucketRequest
{
BucketName = bucketName,
UseClientRegion = true
};
var response = await client.PutBucketAsync(request);
Console.WriteLine("Bucket creation result: {0}", response.HttpStatusCode);
}
And they're available at the very top:


public class S3Manager
{
simple enough (5%)
public static async Task RunAsync(IAmazonS3 client)
{
var bucketName = "sbdemo0610";
await CreateBucketAsync(client, bucketName);
}


File Edit View Navigate Code Refactor Build Run Tests Tools VCS Window Help
it_clawsda22 [C:\work\LLW\it_clawsda22]
Version Control
Search everywhere
it_clawsda22
Run
Debug
Solution
Program.cs
S3Manager.cs
it_clawsda22
Dependencies
Program.cs
S3Manager.cs
Scratches and Consoles
// See https://aka.ms/new-console-template for more information
using ...
Console.WriteLine("Hello, World!");
var client = new AmazonS3Client(awsAccessKeyId: "AKIAXFS7KEM5XA5CFZER", awsSecretAccessKey: "y/1qs/R/xtI
await S3Manager.RunAsync(client);
it_clawsda22
<top-level-entry-point>
TODO
Problems
Unit Tests
NuGet
Dynamic Program Analysis
Terminal
Python Packages
Services
AWSSDK.S3 v3.7.101.44 was successfully installed to it_clawsda22
8:1
CRLF
UTF-8
4 spaces
Good. And now let's go back to the Terminal

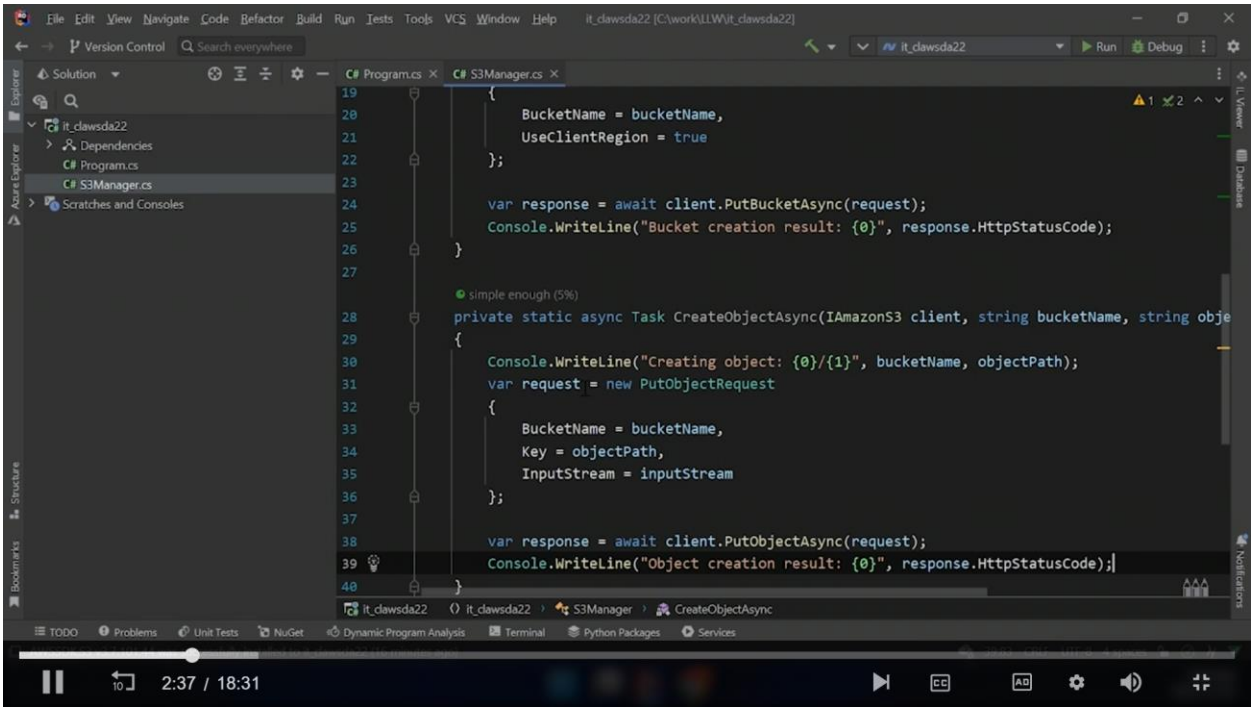

Terminal:
Local
PS C:\work\LLW\it_clawsda22> dotnet run
Hello, World!
PS C:\work\LLW\it_clawsda22> dotnet run
Hello, World!
Creating bucket: sbdemo0610
Bucket creation result: OK

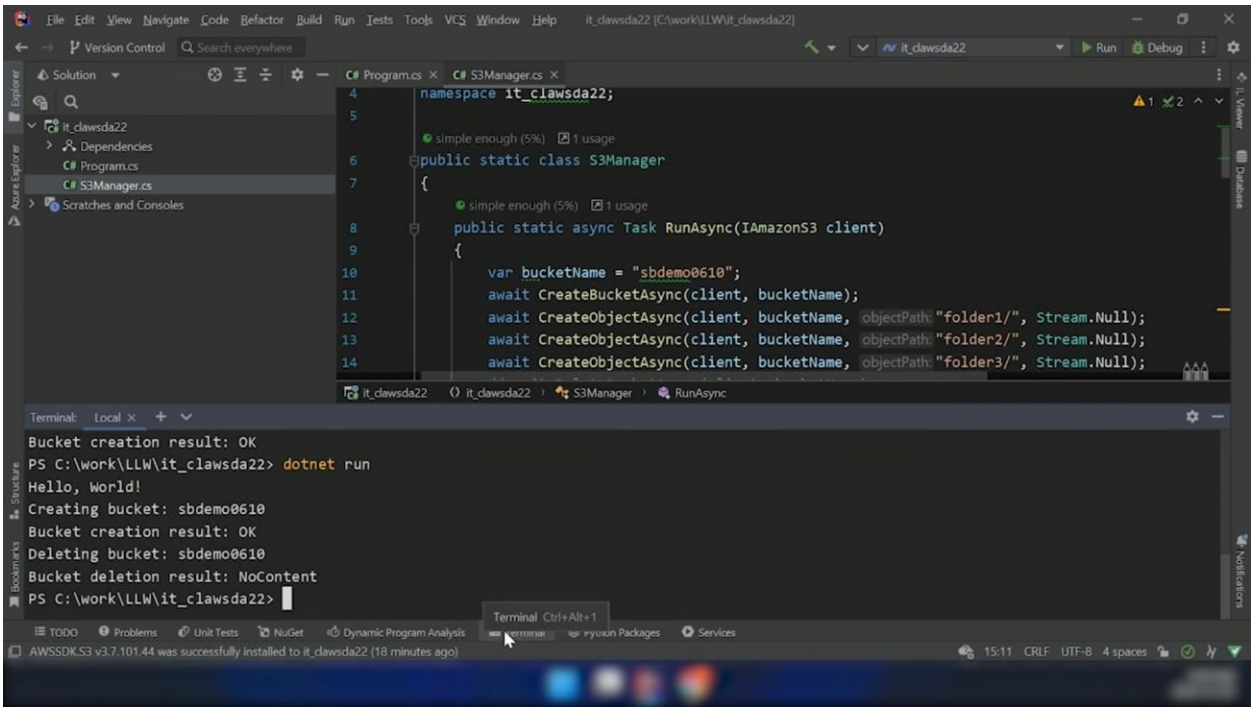
```
simple enough (5%)
private static async Task DeleteBucketAsync(IAmazonS3 client, string bucketName)
{
    Console.WriteLine("Deleting bucket: {0}", bucketName);
    var request = new DeleteBucketRequest
    {
        BucketName = bucketName
    };

    var response = await client.DeleteBucketAsync(request);
    Console.WriteLine("Bucket deletion result: {0}", response.HttpStatusCode);
}
```

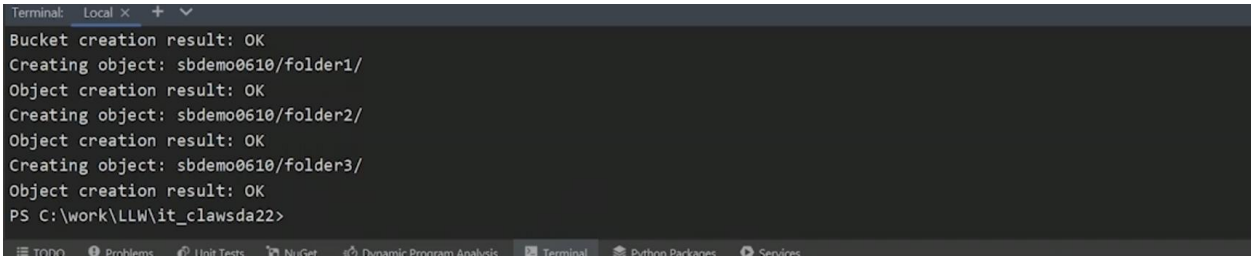
```
public static async Task RunAsync(IAmazonS3 client)
{
    var bucketName = "sbdemo0610";
    await CreateBucketAsync(client, bucketName);
    await DeleteBucketAsync(client, bucketName);
}
```

Creating Amazon S3 Object



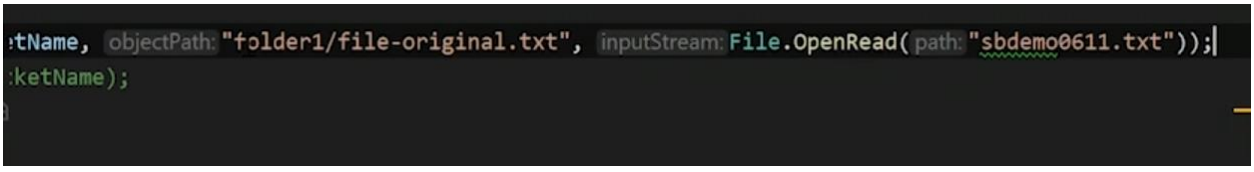


Comment delete await



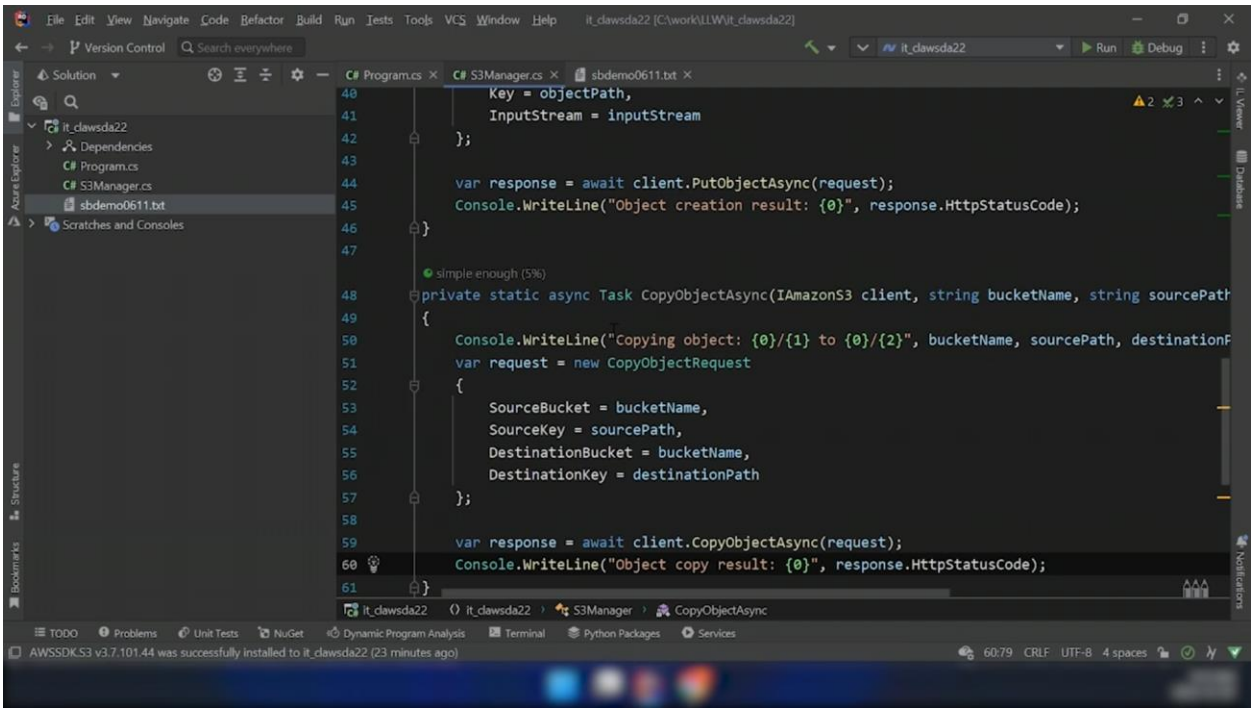
Uploading file in folder

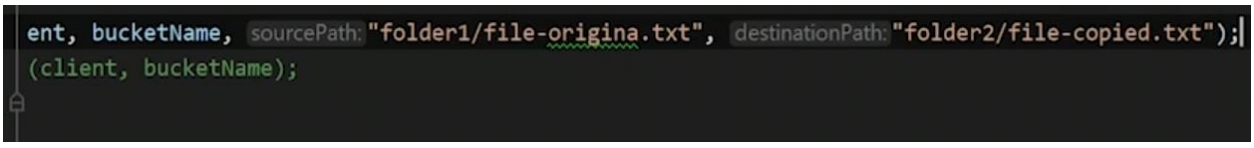
➔ Create a txt file with some content



```
Creating object: sbdemo0610/folder1/file-original.txt
Object creation result: OK
PS C:\work\LLW\it_clawsda22>
```

Copy file





Delete file

```
simple enough (5%)
private static async Task DeleteObjectAsync(IAmazonS3 client, string bucketName, string obje
{
    Console.WriteLine("Deleting object: {0}/{1}", bucketName, objectPath);
    var request = new DeleteObjectRequest
    {
        BucketName = bucketName,
        Key = objectPath
    };

    var response = await client.DeleteObjectAsync(request);
    Console.WriteLine("Object delete result: {0}", response.HttpStatusCode);
}
```

```
await DeleteObjectAsync(client, bucketName, objectPath: "folder3/file-renamed.txt");
//await DeleteBucketAsync(client, bucketName);
```

Move object file

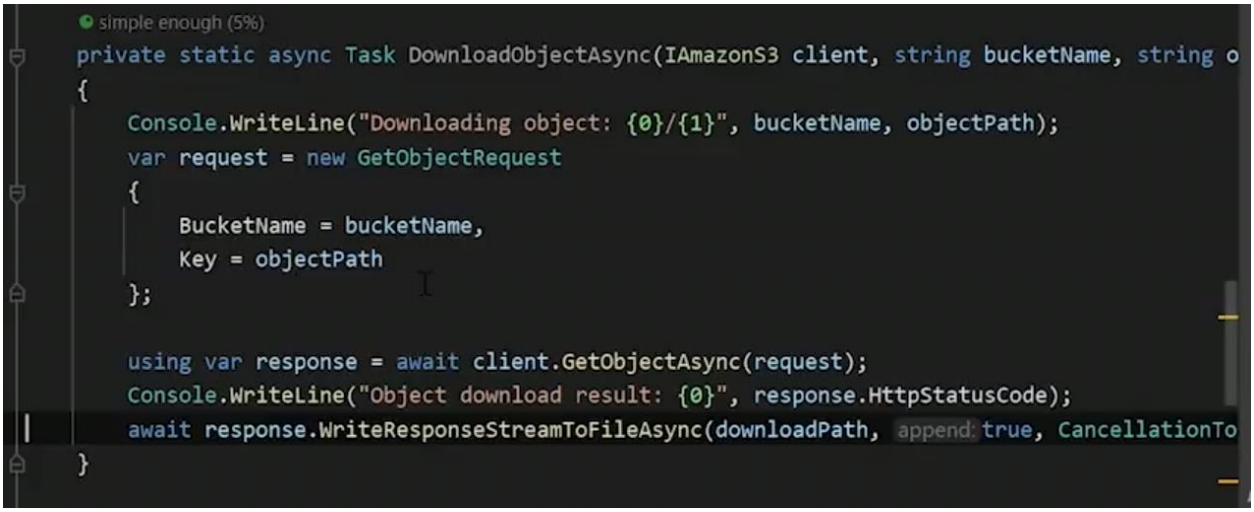
```
private static async Task MoveObjectAsync(IAmazonS3 client, string bucketName, string sourcePath
{
    Console.WriteLine("Moving object: {0}/{1} to {0}/{2}", bucketName, sourcePath, destinationPa
    await CopyObjectAsync(client, bucketName, sourcePath, destinationPath);
    await DeleteObjectAsync(client, bucketName, sourcePath);
}
```

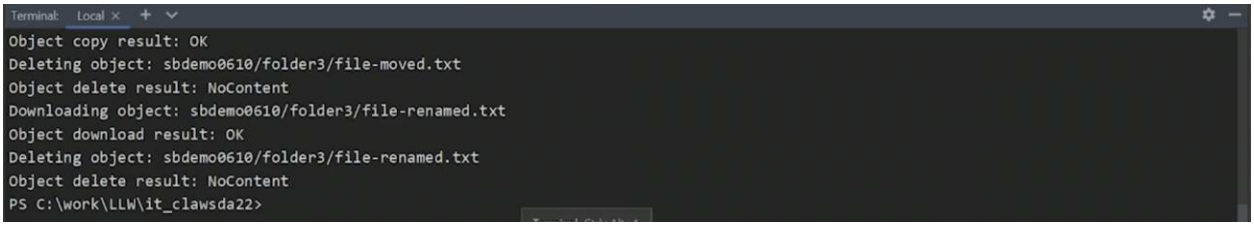
```
.ent, bucketName, sourcePath: "folder2/file-copied.txt", destinationPath: "folder3/file-moved.txt");
```

```
await MoveObjectAsync(client, bucketName, sourcePath: "folder2/file-copied.txt", destinatio
await MoveObjectAsync(client, bucketName, sourcePath: "folder3/file-move")
```

```
bucketName, sourcePath: "folder2/file-copied.txt", destinationPath: "folder3/file-moved.txt");
bucketName, sourcePath: "folder3/file-moved.txt", destinationPath: "folder3/file-renamed.txt");
```

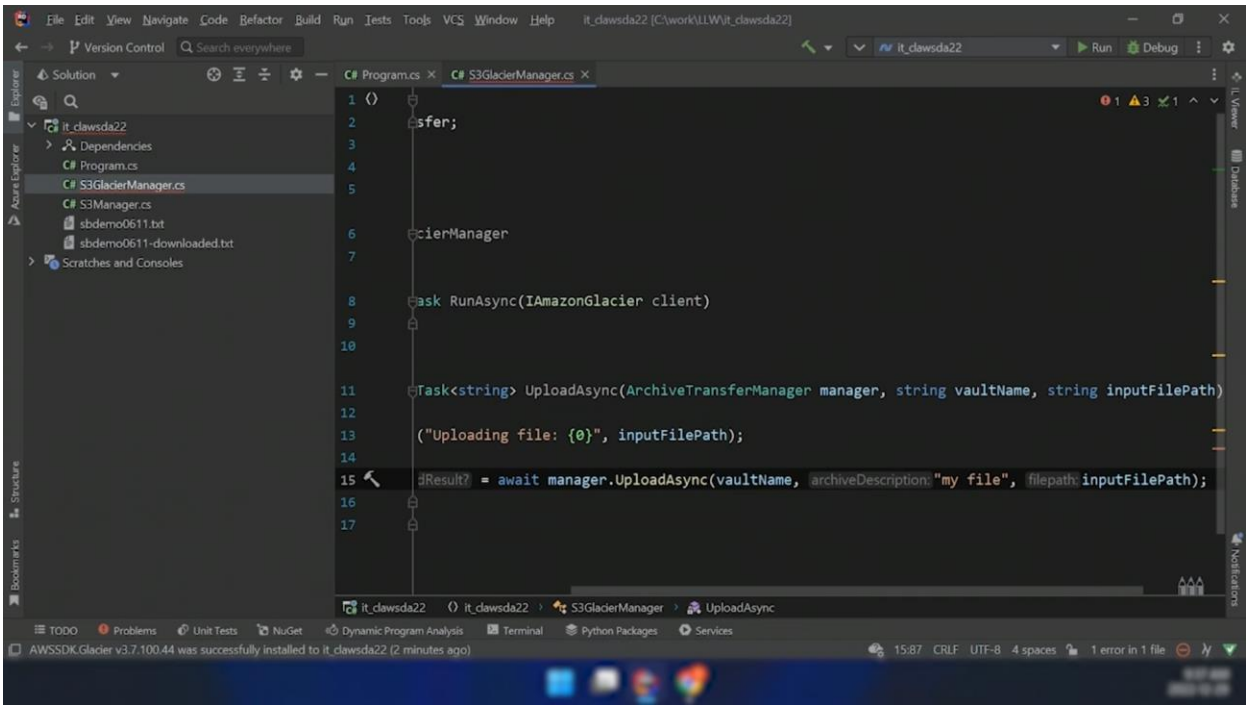
Download file

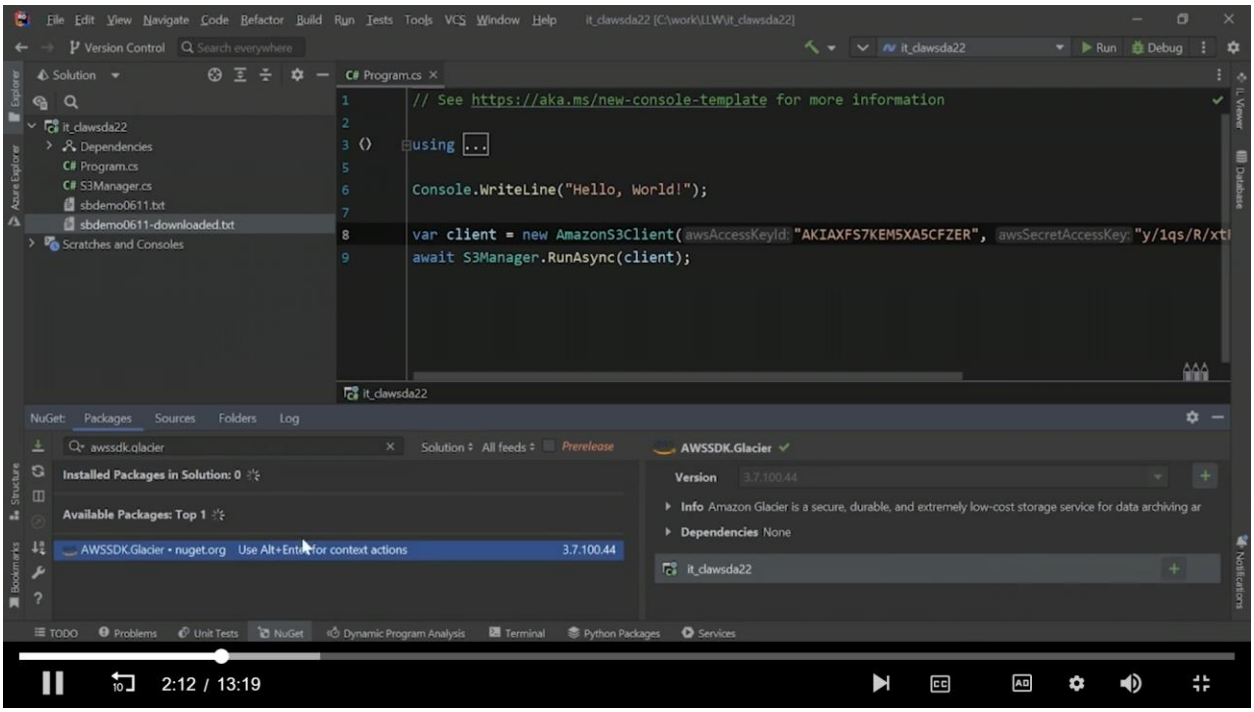


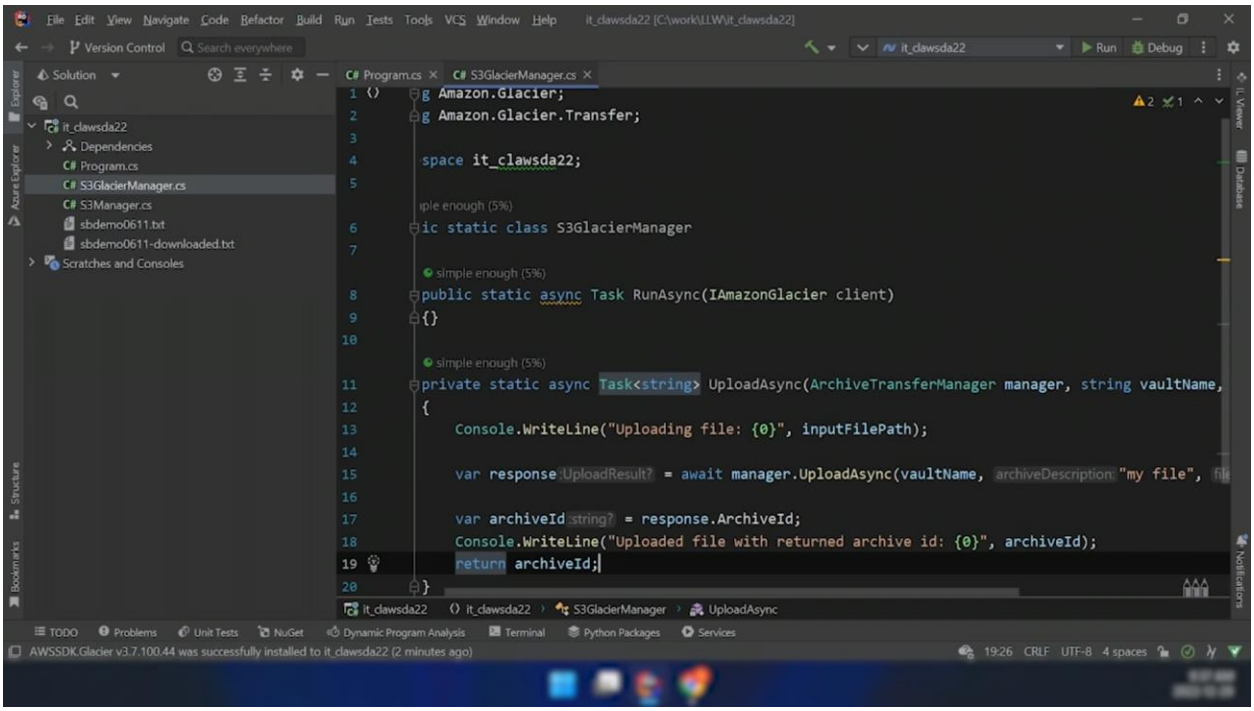


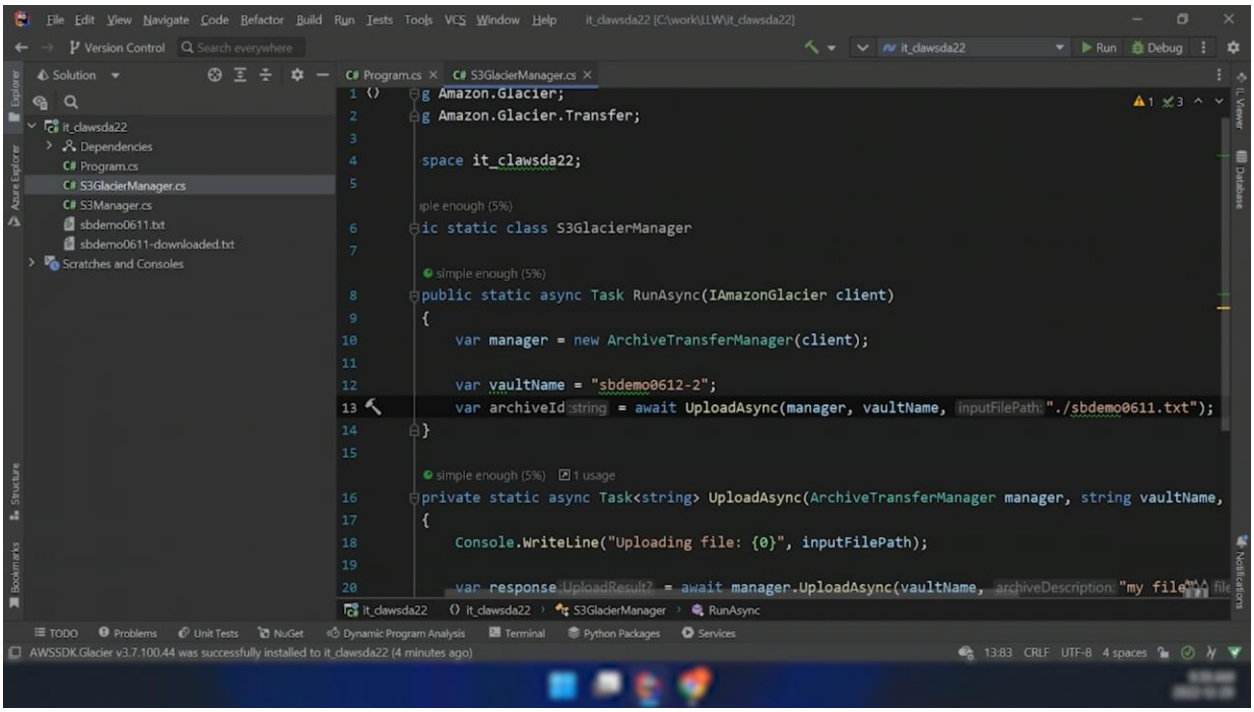
dotnet run = run command on terminal

Working With Amazon S3 Glacier





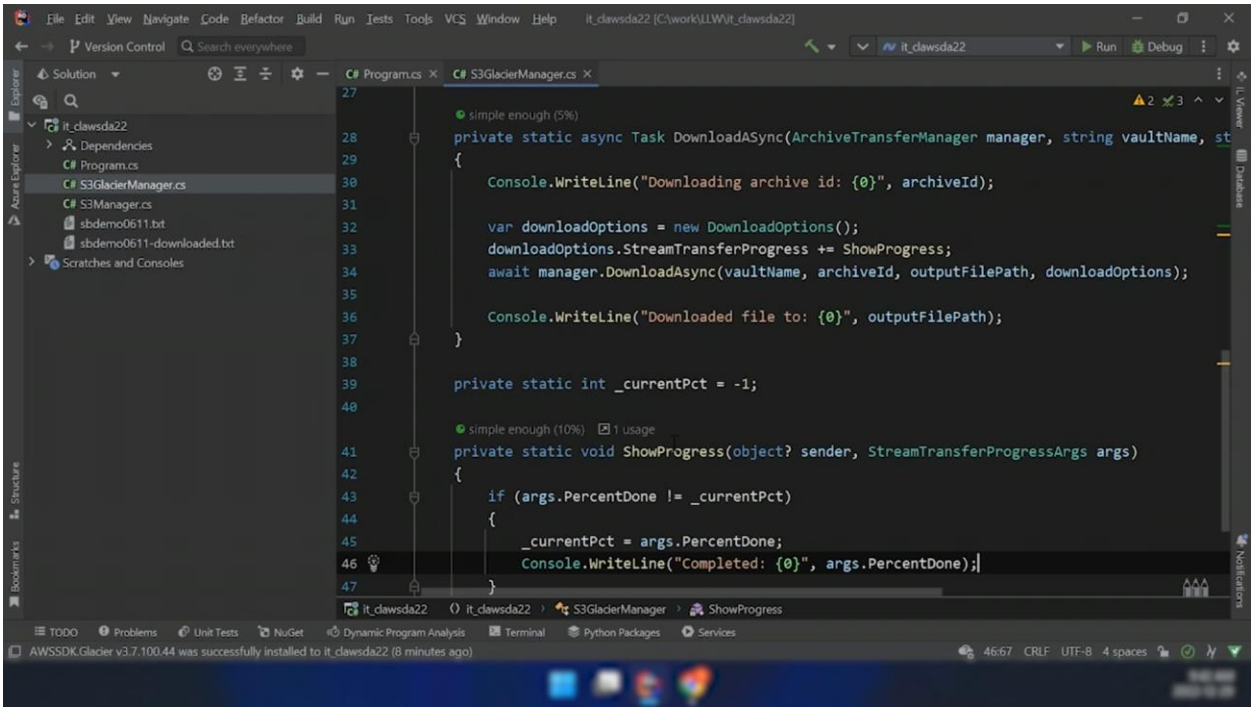

File Edit View Navigate Code Refactor Build Run Tests Tools VCS Window Help
it_clawsda22 [C:\work\LLW\it_clawsda22]
Version Control
Search everywhere
it_clawsda22
Run
Debug
Solution
Program.cs
S3GlacierManager.cs
it_clawsda22
Dependencies
Program.cs
S3GlacierManager.cs
S3Manager.cs
sbdemo0611.txt
sbdemo0611-downloaded.txt
Scratches and Consoles
g Amazon.Glacier;
g Amazon.Glacier.Transfer;
space it_clawsda22;
ple enough (5%)
ic static class S3GlacierManager
simple enough (5%)
public static async Task RunAsync(IAmazonGlacier client)
{}
simple enough (5%)
private static async Task<string> UploadAsync(ArchiveTransferManager manager, string vaultName,
{
Console.WriteLine("Uploading file: {0}", inputFilePath);
var response UploadResult? = await manager.UploadAsync(vaultName, archiveDescription: "my file",
var archiveId string? = response.ArchiveId;
Console.WriteLine("Uploaded file with returned archive id: {0}", archiveId);
return archiveId;
}
it_clawsda22 it_clawsda22 S3GlacierManager UploadAsync
TODO Problems Unit Tests NuGet Dynamic Program Analysis Terminal Python Packages Services
AWSSDK.Glacier v3.7.100.44 was successfully installed to it_clawsda22 (2 minutes ago)
19:26 CRLF UTF-8 4 spaces

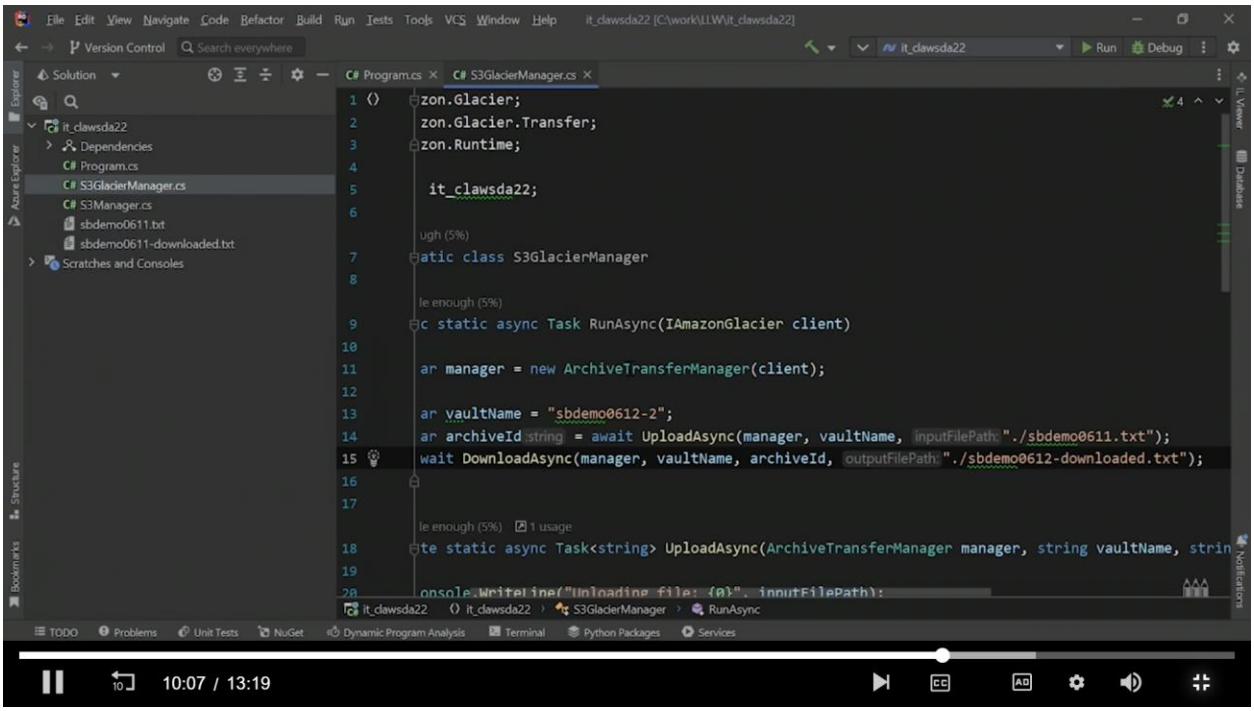

File Edit View Navigate Code Refactor Build Run Tests Tools VCS Window Help
it_clawsda22 [C:\work\LLW\it_clawsda22]
Version Control
Search everywhere
it_clawsda22
Run
Debug
Solution
Program.cs
S3GlacierManager.cs
it_clawsda22
Dependencies
Program.cs
S3GlacierManager.cs
S3Manager.cs
sbdemo0611.txt
sbdemo0611-downloaded.txt
Scratches and Consoles
g Amazon.Glacier;
g Amazon.Glacier.Transfer;
space it_clawsda22;
ple enough (5%)
ic static class S3GlacierManager
simple enough (5%)
public static async Task RunAsync(IAmazonGlacier client)
{
var manager = new ArchiveTransferManager(client);
var vaultName = "sbdemo0612-2";
var archiveId string = await UploadAsync(manager, vaultName, inputFilePath: "./sbdemo0611.txt");
}
simple enough (5%) 1 usage
private static async Task<string> UploadAsync(ArchiveTransferManager manager, string vaultName,
{
Console.WriteLine("Uploading file: {0}", inputFilePath);
var response UploadResult? = await manager.UploadAsync(vaultName, archiveDescription: "my file
it_clawsda22 it_clawsda22 S3GlacierManager RunAsync
TODO Problems Unit Tests NuGet Dynamic Program Analysis Terminal Python Packages Services
AWSSDK.Glacier v3.7.100.44 was successfully installed to it_clawsda22 (4 minutes ago)
13:83 CRLF UTF-8 4 spaces

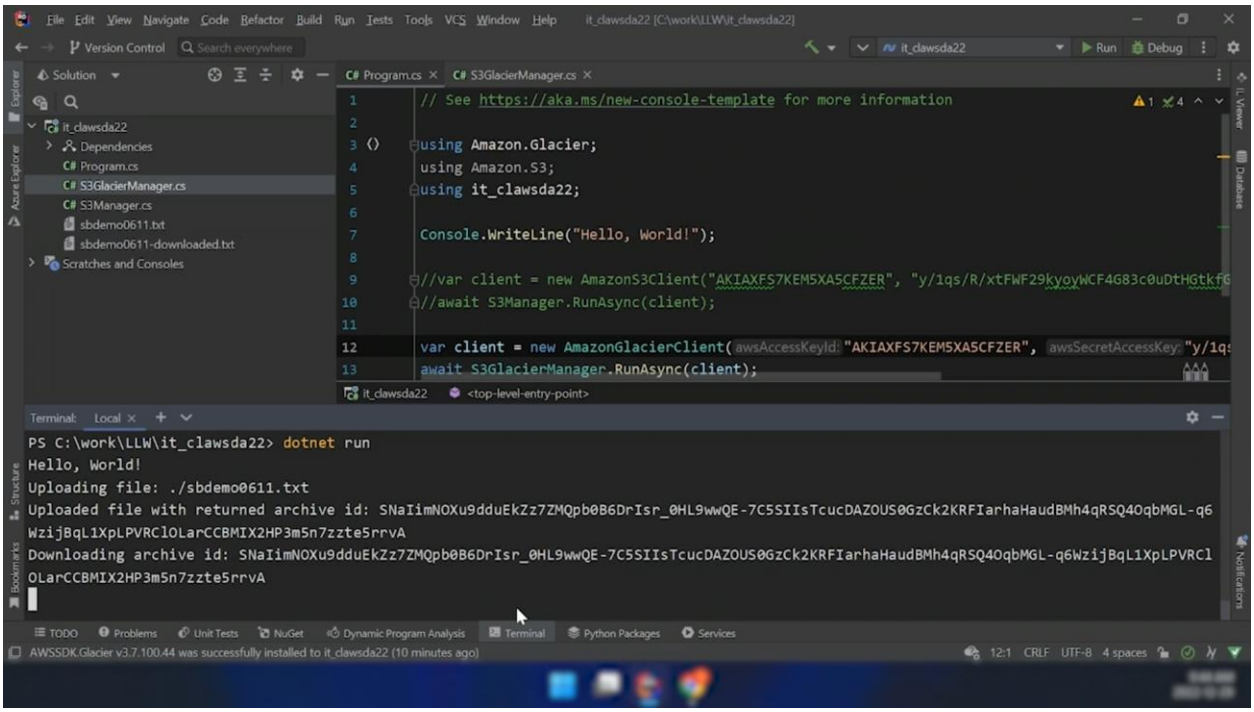
```
simple enough (5%)
private static async Task DownloadASync(ArchiveTransferManager manager, string vaultName, s
{
    Console.WriteLine("Downloading archive id: {0}", archiveId);

    var downloadOptions = new DownloadOptions();
    downloadOptions.StreamTransferProgress += ShowProgress;
    await manager.DownloadAsync(vaultName, archiveId, outputFilePath, downloadOptions);

    Console.WriteLine("Downloaded file to: {0}", outputFilePath);
}
```

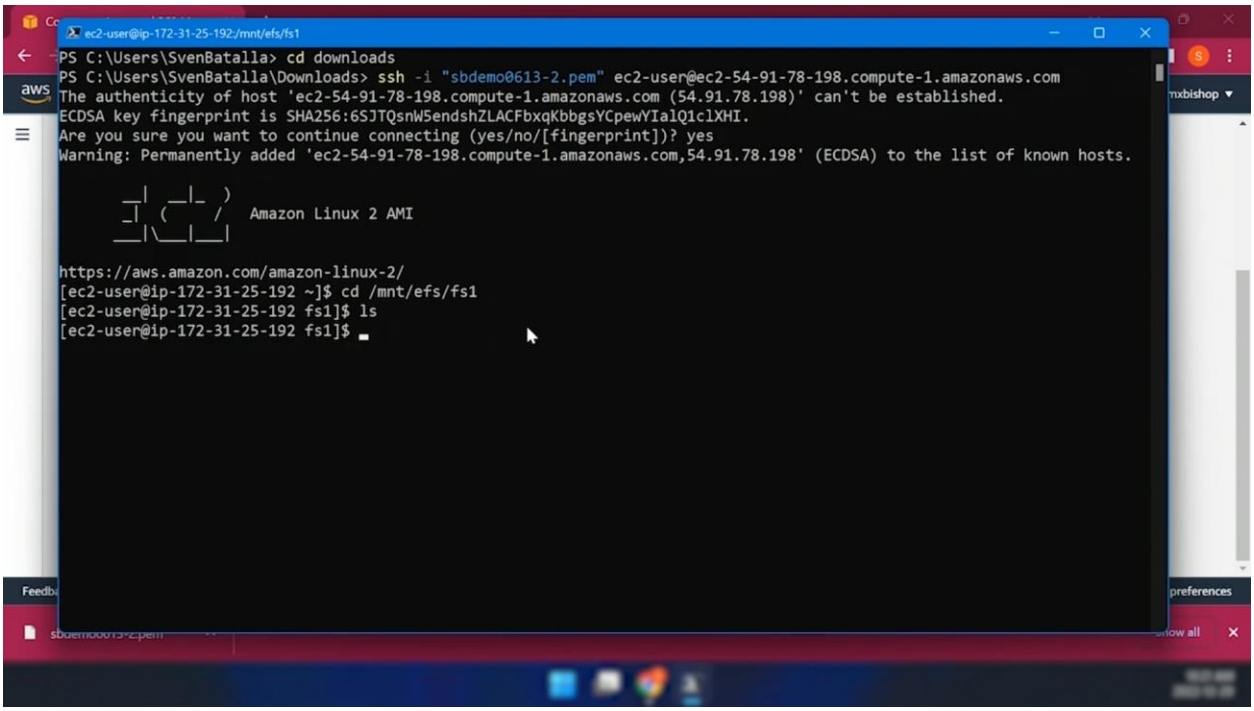






Creating an Amazon EFS File System

SSH client should be installed in pc that comes along with windows 11 and you have to enable it



1. Which are other terms for an AWS Backup backup?

- **Recovery point**
- **Snapshot**

2. Which character is required at the end of an object name to tell Amazon S3 to create a folder?

- **Forward slash (/)**

3. Which encryption strategy requires the application to perform the encryption of data before it is written to Amazon S3?

- **Client-side**

4. Which are examples of items that can be placed in an Amazon S3 bucket?

- **Objects**
- **Folders**

5. Which IP protocols are supported by dual-stack endpoints?
- **IPv4**
- **IPv6**

6. Which is the top-level component of Amazon S3?
- **Buckets**

7. Which Amazon S3 Glacier tier supports choosing between paying for access to archives within 1-5 minutes or free access between 5-12 hours?
- **S3 Glacier Flexible Retrieval**

8. Which Amazon EFS storage class takes advantage of multiple availability zones to offer high availability and self-healing?
- **Standard storage classes**

9. By default, which location is an EFS file system mounted to a Linux EC2 system?
- **/mnt/efs/fs1**

10. Which two credential values are required when establishing a connection to AWS using the AWS SDK?
- **Key ID**
- **Secret value**

11. Which Amazon EBS volume type has the highest performance?
- **Solid state drives (SSDs)**

12. Which value is required to interact with an archive after it has been created?
- **Archive ID**