Quickstart

This guide details the steps needed to install or update the AWS SDK for Python.

The SDK is composed of two key Python packages: Botocore (the library providing the low-level functionality shared between the Python SDK and the AWS CLI) and Boto3 (the package implementing the Python SDK itself).

> **Note**
>
> Documentation and developers tend to refer to the AWS SDK for Python as "Boto3," and this documentation often does so as well.

Installation

To use Boto3, you first need to install it and its dependencies.

Install or update Python

Before installing Boto3, install Python 3.8 or later; support for Python 3.7 and earlier is deprecated. After the deprecation date listed for each Python version, new releases of Boto3 will not include support for that version of Python. For details, including the deprecation schedule and how to update your project to use Python 3.8, see Migrating to Python 3.

For information about how to get the latest version of Python, see the official Python documentation.

Install Boto3

Install the latest Boto3 release via **pip**:

```
pip install boto3
```

If your project requires a specific version of Boto3, or has compatibility concerns with certain versions, you may provide constraints when installing:

```
# Install Boto3 version 1.0 specifically
pip install boto3==1.0.0

# Make sure Boto3 is no older than version 1.15.0
pip install boto3>=1.15.0

# Avoid versions of Boto3 newer than version 1.15.3
pip install boto3<=1.15.3
```

> **Note**
>
> The latest development version of Boto3 is on GitHub.

Using the AWS Common Runtime (CRT)

In addition to the default install of Boto3, you can choose to include the new AWS Common Runtime (CRT). The AWS CRT is a collection of modular packages that serve as a new foundation for AWS SDKs. Each library provides better performance and minimal footprint for the functional area it implements. Using the CRT, SDKs can share the same base code when possible, improving consistency and throughput optimizations across AWS SDKs.

When the AWS CRT is included, Boto3 uses it to incorporate features not otherwise available in the AWS SDK for Python.

You'll find it used in features like:

- Amazon S3 Multi-Region Access Points
- Amazon S3 Object Integrity
- Amazon EventBridge Global Endpoints

However, Boto3 doesn't use the AWS CRT by default but you can opt into using it by specifying the `crt` extra feature when installing Boto3:

```
pip install boto3[crt]
```

To revert to the non-CRT version of Boto3, use this command:

```
pip uninstall awscrt
```

If you need to re-enable CRT, reinstall `boto3[crt]` to ensure you get a compatible version of `awscrt`:

```
pip install boto3[crt]
```

Configuration

Before using Boto3, you need to set up authentication credentials for your AWS account using either the IAM Console or the AWS CLI. You can either choose an existing user or create a new one.

For instructions about how to create a user using the IAM Console, see Creating IAM users. Once the user has been created, see Managing access keys to learn how to create and retrieve the keys used to authenticate the user.

If you have the AWS CLI installed, then you can use the **aws configure** command to configure your credentials file:

```
aws configure
```

Alternatively, you can create the credentials file yourself. By default, its location is `~/.aws/credentials`. At a minimum, the credentials file should specify the access key and secret access key. In this example, the key and secret key for the account are specified in the `default` profile:

```
[default]
aws_access_key_id = YOUR_ACCESS_KEY
aws_secret_access_key = YOUR_SECRET_KEY
```

You may also want to add a default region to the AWS configuration file, which is located by default at `~/.aws/config`:

```
[default]
region=us-east-1
```

Alternatively, you can pass a `region_name` when creating clients and resources.

You have now configured credentials for the default profile as well as a default region to use when creating connections. See Configuration for in-depth configuration sources and options.

Using Boto3

To use Boto3, you must first import it and indicate which service or services you're going to use:

```
import boto3

# Let's use Amazon S3
s3 = boto3.resource('s3')
```

Now that you have an `s3` resource, you can make send requests to the service. The following code uses the `buckets` collection to print out all bucket names:

```
# Print out bucket names
for bucket in s3.buckets.all():
    print(bucket.name)
```

You can also upload and download binary data. For example, the following uploads a new file to S3, assuming that the bucket `amzn-s3-demo-bucket` already exists:

```
# Upload a new file
with open('test.jpg', 'rb') as data:
    s3.Bucket('amzn-s3-demo-bucket').put_object(Key='test.jpg', Body=data)
```

Resources and Collections are covered in more detail in the following sections.