AWS SQS and SNS

### Simple Queue Service (SQS) Overview

Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. SQS eliminates the complexity and overhead associated with managing and operating message oriented middleware, and empowers developers to focus on differentiating work. Using SQS, you can send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available.

### Use cases / Considerations

Amazon Simple Queue Service (Amazon SQS) offers a secure, durable, and available hosted queue that lets you integrate and decouple distributed software systems and components. Amazon SQS offers common constructs such as dead-letter queues and cost allocation tags. It provides a generic web services API that you can access using any programming language that the AWS SDK supports.

Amazon SQS supports both standard and FIFO queues.

* Amazon SQS offers standard as the default queue type. Standard queues support a nearly unlimited number of API calls per second, per API action (SendMessage, ReceiveMessage, or DeleteMessage). Standard queues support at-least-once message delivery. However, occasionally (because of the highly distributed architecture that allows nearly unlimited throughput), more than one copy of a message might be delivered out of order. Standard queues provide best-effort ordering which ensures that messages are generally delivered in the same order as they're sent.
* FIFO (First-In-First-Out) queues are designed to enhance messaging between applications when the order of operations and events is critical, or where duplicates can't be tolerated.

More about standart queues.

You can use standard message queues in many scenarios, as long as your application can process messages that arrive more than once and out of order.

For example:

* Decouple live user requests from intensive background work – Let users upload media while resizing or encoding it.
* Allocate tasks to multiple worker nodes – Process a high number of credit card validation requests.
* Batch messages for future processing – Schedule multiple entries to be added to a database.

*Note:*

*A standard queue makes a best effort to preserve the order of messages, but more than one copy of a message might be delivered out of order.*

At-least-once delivery.

Amazon SQS stores copies of your messages on multiple servers for redundancy and high availability. On rare occasions, one of the servers that stores a copy of a message might be unavailable when you receive or delete a message.

If this occurs, the copy of the message isn't deleted on that unavailable server, and you might get that message copy again when you receive messages. Design your applications to be idempotent (they should not be affected adversely when processing the same message more than once).

More about FIFO queues.

Where is FIFO the best choose:

* To make sure that user-entered commands are run in the right order.
* To display the correct product price by sending price modifications in the right order.
* To prevent a student from enrolling in a course before registering for an account.

FIFO queues also provide exactly-once processing but have a limited number of transactions per second (TPS).

*Note:*

*The name of a FIFO queue must end with the .fifo suffix. The suffix counts towards the 80-character queue name quota. To determine whether a queue is FIFO, you can check whether the queue name ends with the suffix.*

FIFO message ordering.

The FIFO queue improves upon and complements the standard queue. The most important features of this queue type are FIFO (First-In-First-Out) delivery and exactly-once processing:

The order in which messages are sent and received is strictly preserved and a message is delivered once and remains available until a consumer processes and deletes it.

Duplicates aren't introduced into the queue.

In addition, FIFO queues support message groups that allow multiple ordered message groups within a single queue. There is no quota to the number of message groups within a FIFO queue.

Amazon SQS dead-letter queues

Amazon SQS supports dead-letter queues, which other queues (source queues) can target for messages that can't be processed (consumed) successfully. Dead-letter queues are useful for debugging your application or messaging system because they let you isolate problematic messages to determine why their processing doesn't succeed.

*Important:*

*The dead-letter queue of a FIFO queue must also be a FIFO queue. Similarly, the dead-letter queue of a standard queue must also be a standard queue.*

Benefits of dead-letter queues.

* Configure an alarm for any messages delivered to a dead-letter queue.
* Examine logs for exceptions that might have caused messages to be delivered to a dead-letter queue.
* Analyze the contents of messages delivered to a dead-letter queue to diagnose software or the producer’s or consumer’s hardware issues.
* Determine whether you have given your consumer sufficient time to process messages.

### General useful links:

* What is Amazon Simple Queue Service

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html>

* Introducing Amazon Simple Queue Service (SQS) FIFO Queues – Messaging on AWS:

<https://www.youtube.com/watch?v=XrX7rb6M3jw&ab_channel=AmazonWebServices>

* Amazon SQS dead-letter queues

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-dead-letter-queues.html>

### Simple Notification Service (SNS) Overview

Amazon Simple Notification Service (Amazon SNS) is a fully managed messaging service for both application-to-application (A2A) and application-to-person (A2P) communication.

SNS provides message delivery from publishers to subscribers (also known as producers and consumers). Publishers communicate asynchronously with subscribers by sending messages to a topic, which is a logical access point and communication channel. Clients can subscribe to the SNS topic and receive published messages using a supported endpoint type, such as Amazon Kinesis Data Firehose, Amazon SQS, AWS Lambda, HTTP, email, mobile push notifications, and mobile text messages (SMS).

Amazon SNS provides the following features and capabilities:

* Application-to-application messaging
* Application-to-person notifications
* Standard and FIFO topics
* Message durability
* Message archiving and analytics
* Message attributes
* Message filtering
* Message filtering

You can use the following services with Amazon SNS:

* Amazon SQS
* AWS Lambda
* AWS Identity and Access Management (IAM)
* AWS CloudFormation

Pricing for Amazon SNS

Amazon SNS has no upfront costs. You pay based on the number of messages that you publish, the number of notifications that you deliver, and any additional API calls for managing topics and subscriptions. Delivery pricing varies by endpoint type. You can get started for free with the Amazon SNS free tier.

### General useful links:

* Amazon SNS

<https://docs.aws.amazon.com/sns/latest/dg/welcome.html>

* Event-Driven Pub-Sub with SNS and SQS

<https://www.youtube.com/watch?v=c_WNBmEc6EE&ab_channel=AmazonWebServices>