Simple Queue Service (SQS)

Fully managed queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications

What is a Queueing System?

What is a Messaging System?
Used to provide asynchronous communication and decouple processes via messages / events
From a sender and receiver (producer and consumer)

Queueing

Generally will delete messages once they are consumed. Simple communication. **Not Real-time**
Have to pull. Not reactive.

VS

Streaming

Multiple consumers can **react** to events (messages), Event live in the stream for long periods of time, so complex operations can be applied. **Real-time**

SQS

Sidekiq

RabbitMQ

Kinesis

APACHE kafka

NATS

Simplified example because there are lots of ifs and buts

Introduction to SQS

SQS is for **Application Integration**.

AWS SQS is a solution for the distributed **queuing** of messages generated by your application. It connects isolate applications together by passing along message to one another.

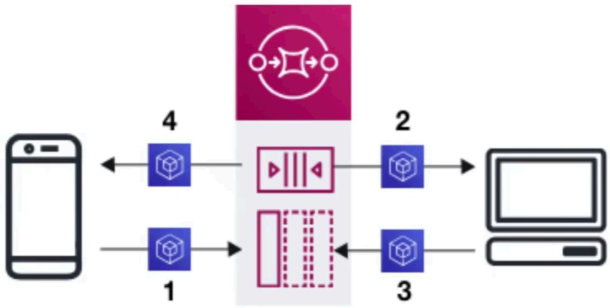
A **queue** is a temporary repository for messages that are awaiting processing.

Using the **AWS SDK** you write code which publishes messages onto the queue or you pull the queue for messages.

SQS is pull based
Not push based!

SQS – Use Case

1. App publishes messages to the queue
2. Other app pulls the queue and find the message and does something
3. Other app reports that they completed their task and marks the message for completion
4. Original app pulls the queue and sees the message is no longer in the queue.



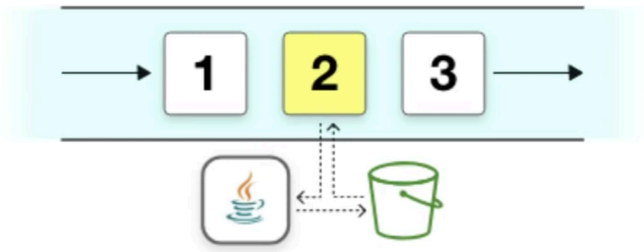
Both apps are using the **AWS SDK** to push messages and pull the queue.

SQS Limits – Message Size

Message Size

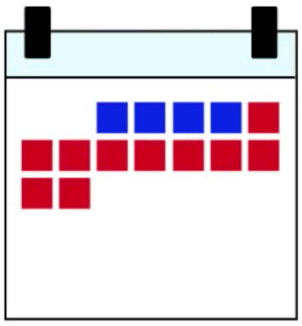
The size of a message can be between **1 byte and 256 KB**

Amazon SQS Extended Client Library for Java lets you send messages **256KB to 2GB** in size. The message will be stored in S3 and library will reference the S3 object.



SQS Limits – Message Retention

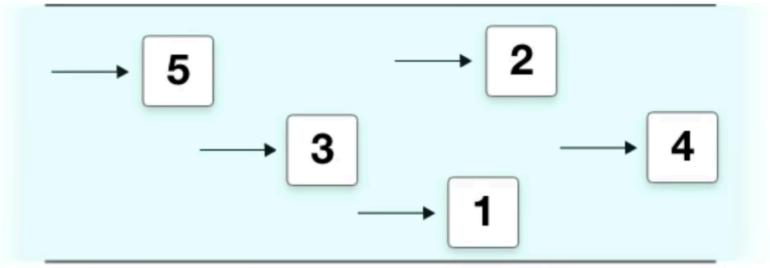
Message Retention
how long SQS will hold onto a message in the queue before dropping it from the queue (deleting it)



Message retention by default is **4 days**.

Message retention can be adjusted from a minimum of **60 seconds** to a max **of 14 Days**

SQS – Standard Queues



AWS SQS **Standard Queues** allow you a **nearly-unlimited** number of transactions per second.

Guarantees that a message will be delivered **AT LEAST once**.

More than one copy of a message could be potentially delivered **out of order**.

Provides **best-effort ordering** that helps ensure a message is generally delivered in the same order that it was sent.

SQS – FIFO Queues

AWS SQS **First-In-First-Out** queues support multiple ordered message groups within a single queue.

Limited to 300 transactions per second.

SWS FIFO queues have all the same capabilities of a Standard Queue

SQS – Visibility Timeout

How do we prevent another app from reading a message while another one Is busy with that message? (Avoid someone doing the same task)

A **visibility time-out** is the period of time that messages are **invisible in the SQS queue**, after a reader picks up that message.

Messages will be **deleted** from the queue **after a job has processed**. (before the visibility timeout expires)

If a job is **NOT** processed before the visibility time-out period, the message will **become visible again** and another reader will process it.

This can result in the same message being delivered **twice**!

30 seconds (default)
0 seconds minimum
12 hours maximum

SQS – Short vs Long Polling

Polling is the method in which we retrieve messages from the queues.

Short polling (default) returns messages immediately, even if the message queue being polled is empty.

When you need a message **right away**. shorting polling is what you want to use.

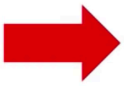
Long polling waits until message **arrives in the queue**, or the **long poll timeout expires**.

Long polling makes it **inexpensive to retrieve messages** from your queue as soon as the messages are available.

Using long polling will reduce the cost because you can **reduce the number of empty receives**.

Most use-cases you want to use Long Polling

You can enable long polling when receiving a message by setting the wait time in seconds on the **ReceiveMessageRequest**



```
1  ReceiveMessageRequest receive_request = new ReceiveMessageRequest()
2  .withQueueUrl(queue_url)
3  .withWaitTimeSeconds(40);
4  sqs.receiveMessage(receive_request);
```

Simple Queue Service (SQS) *CheatSheet*

- SQS is a queuing service using messages with a queue. Think Sidekiq or RabbitMQ
- SQS is used for Application Integration, it lets decoupled services and apps to talk to each other
- To read SQS use need to **pull** the queue using the AWS SDK. SQS is **not pushed-based**
- SQS supports both Standard and First-In-First-Out (FIFO) queues
- Standard allows nearly unlimited messages per second, does not guarantee order of delivery, always delivers at least once, you must protect again duplicate messages being processed
- FIFO maintain the order of messages with a 300 limit
- There are two kinds of polling Short (Default) and Long Polling
- Short polling returns messages immediately, even if the message queue being polled is empty.
- Long polling waits until message arrives in the queue, or the long poll timeout expires.
- In majority of cases Long polling is preferred over short polling.
- **Visibility time-out** is the period of time that messages are invisible in the SQS queue
- Messages will be deleted from queue after a job has processed. (before visibility timeout expires)
- If Visibility timeout expires than a job will become visible to the queue
- The default Visibility time-out is 30 seconds. Timeout can be **0 seconds** to a maximum of 12 hours.
- SQS can retain messages from 60 seconds to 14 days and by default is 4 days
- Message size between 1 byte to 256 kb, Extended Client Library for Java can increase to 2GB