# NATS CLI Messaging and JetStream

**Publisher** nats pub → subject → **Server** routes by subject → captured by → **Stream** JetStream, stored → consumer reads → **Consumer** cursor + acks

## CONCEPTS

Subjects are dot-separated tokens. * matches one token, > matches all remaining tokens.

Plain pub/sub is fire and forget: a subscriber that is offline misses the message. Streams (JetStream) keep it.

JetStream must be enabled on the server (nats-server -js), otherwise stream, consumer, kv and obj fail with no responders.

## CONNECT AND CONTEXTS

**nats context add** *<name>* **--server** *<url>* **--select**
Save a server as a context

**nats context ls**
List saved contexts

**nats context select** *<name>*
Switch context

**nats --server** *<url>* *<command>*
Use a server without a context

## CONNECT AND CONTEXTS (cont.)

**nats --creds** *<file>* *<command>*
Authenticate with a .creds file

**NATS_URL=***<url>* **nats** *<command>*
Set the server by environment

## PUBLISH AND SUBSCRIBE

**nats pub** *<subject>* "*<msg>*"
Publish a message

**nats sub** *<subject>*
Print messages as they arrive

**nats sub "***<prefix>***.>"**
Subscribe to a whole subject tree

**nats sub** *<subject>* **--queue** *<group>*
Share messages across a queue group

**echo "***<msg>***" | nats pub** *<subject>*
Publish from stdin

**nats pub** *<subject>* "*<msg>*" **--jetstream**
Publish and wait for a stream ack

## REQUEST AND REPLY

**nats request** *<subject>* "*<msg>*"
Send a request, print the reply

**nats reply "***<subject>***"** "*<body>*"
Answer requests with a fixed body

**nats reply "***<subject>***" --command "***<cmd>***"**
Answer with a command's output

## SERVER AND ACCOUNT

**nats account info**
Show account limits and usage

**nats-server -js**
Start a local server with JetStream

## STREAMS

**nats stream add** *<name>* **--subjects "***<s>***.>" --defaults**
Create a stream

**nats stream ls**
List streams

**nats stream info** *<name>*
Show config and state

## STREAMS (cont.)

**nats stream view** *<name>*
Browse stored messages

**nats stream rm** *<name>*
Delete a stream

**nats stream purge** *<name>*
Remove all messages

## CONSUMERS

**nats consumer add** *<stream>* *<consumer>* **--pull --defaults**
Create a durable pull consumer

**nats consumer next** *<stream>* *<consumer>* **--count** *<n>*
Fetch and ack n messages

**nats consumer ls** *<stream>*
List a stream's consumers

**nats consumer info** *<stream>* *<consumer>*
Show pending and ack state

## KEY-VALUE STORE

**nats kv add** *<bucket>*
Create a bucket

**nats kv put** *<bucket>* *<key>* *<value>*
Set a key

## KEY-VALUE STORE (cont.)

**nats kv get** *<bucket>* *<key>*
Read a key

**nats kv ls**
List buckets

**nats kv watch** *<bucket>*
Stream changes live

**nats kv ls** *<bucket>*
List keys in a bucket

**nats kv del** *<bucket>* *<key>*
Delete a key (leaves a marker)

**nats kv history** *<bucket>* *<key>*
Show all revisions of a key

## OBJECT STORE

**nats object get** *<bucket>* *<name>*
Download a file

**nats object put** *<bucket>* *<file>*
Upload a file

## BENCHMARK

**nats bench pub** *<subject>* **--msgs** *<n>* **--clients** *<n>*
Measure publish throughput