
Bunny, a Ruby RabbitMQ Client

Bunny is a RabbitMQ client that focuses on ease of use. It is feature complete, supports all recent RabbitMQ features and does not have any heavyweight dependencies.

I Know What RabbitMQ and Bunny are, How Do I Get Started?

Right here!

What is Bunny Good For?

One can use Bunny to make Ruby applications interoperate with other applications (both built in Ruby and not). Complexity and size may vary from simple work queues to complex multi-stage data processing workflows that involve many applications built with all kinds of technologies.

Specific examples:

- Events collectors, metrics & analytics applications can aggregate events produced by various applications (Web and not) in the company network.
- A Web application may route messages to a Java app that works with SMS delivery gateways.
- MMO games can use flexible routing RabbitMQ provides to propagate event notifications to players and locations.
- Price updates from public markets or other sources can be distributed between interested parties, from trading systems to points of sale in a specific geographic region.
- Content aggregators may update full-text search and geospatial search indexes by delegating actual indexing work to other applications over RabbitMQ.
- Companies may provide streaming/push APIs to their customers, partners or just general public.
- Continuous integration systems can distribute builds between multiple machines with various hardware and software configurations using advanced routing features of RabbitMQ.
- An application that watches updates from a real-time stream (be it markets data or Twitter stream) can propagate updates to interested parties, including Web applications that display that information in the real time.

Supported Ruby Versions

Modern Bunny versions support

- CRuby 2.6 through 3.1 (inclusive)
- TruffleRuby

For environments that use TLS, Bunny expects Ruby installations to use a recent enough OpenSSL version that **includes support for TLS 1.3**.

JRuby

Bunny works sufficiently well on JRuby but there are known JRuby bugs in versions prior to JRuby 9000 that cause high CPU burn. JRuby users should use March Hare.

Bunny 1.7.x was the last version to support CRuby 1.9.3 and 1.8.7

Supported RabbitMQ Versions

Modern Bunny releases target currently supported RabbitMQ release series.

Change Log

Bunny is a mature library (started in early 2009) with a stable public API.

Change logs per release series:

- main (most notable changes for all release series)
- 2.19.x

Installation & Bundler Dependency

Most Recent Release

build unknown

Bundler Dependency

To use Bunny in a project managed with Bundler:

```
1 gem "bunny", ">= 2.19.0"
```

With Rubygems

To install Bunny with RubyGems:

```
1 gem install bunny
```

Quick Start

Below is a small snippet that demonstrates how to publish and synchronously consume (“pull API”) messages with Bunny.

For a 15 minute tutorial using more practical examples, see [Getting Started with RabbitMQ and Ruby using Bunny](#).

```
1 require "bunny"
2
3 # Start a communication session with RabbitMQ
4 conn = Bunny.new
5 conn.start
6
7 # open a channel
8 ch = conn.create_channel
9 ch.confirm_select
10
11 # declare a queue
12 q = ch.queue("test1")
13 q.subscribe(manual_ack: true) do |delivery_info, metadata, payload|
14   puts "This is the message: #{payload}"
15   # acknowledge the delivery so that RabbitMQ can mark it for deletion
16   ch.ack(delivery_info.delivery_tag)
17 end
18
19 # publish a message to the default exchange which then gets routed to
   this queue
20 q.publish("Hello, everybody!")
21
22 # await confirmations from RabbitMQ, see
23 # https://www.rabbitmq.com/publishers.html#data-safety for details
24 ch.wait_for_confirms
25
26 # give the above consumer some time consume the delivery and print out
   the message
27 sleep 1
28
29 puts "Done"
30
31 ch.close
32 # close the connection
```

Documentation

Getting Started

For a 15 minute tutorial using more practical examples, see [Getting Started with RabbitMQ and Ruby using Bunny](#).

Guides

Bunny documentation guides are under [docs/guides](#) in this repository:

- [Queues and Consumers](#)
- [Exchanges and Publishers](#)
- [AMQP 0.9.1 Model Explained](#)
- [Connecting to RabbitMQ](#)
- [Error Handling and Recovery](#)
- [TLS/SSL Support](#)
- [Bindings](#)
- [Using RabbitMQ Extensions with Bunny](#)
- [Durability and Related Matters](#)

Some highly relevant RabbitMQ documentation guides:

- [Connections](#)
- [Channels](#)
- [Queues](#)
- [Quorum queues](#)
- [Streams](#) (Bunny can perform basic operations on streams even though it does not implement the RabbitMQ Stream protocol)
- [Publishers](#)
- [Consumers](#)
- [Data safety: publisher and consumer Confirmations](#)
- [Production Checklist](#)

API Reference

[Bunny API Reference](#).

Community and Getting Help

Mailing List

Bunny has a mailing list. Please use it for all questions, investigations, and discussions. GitHub issues should be used for specific, well understood, actionable maintainers and contributors can work on.

We encourage you to also join the RabbitMQ mailing list mailing list. Feel free to ask any questions that you may have.

Continuous Integration

build unknown

Reporting Issues

If you find a bug you understand well, poor default, incorrect or unclear piece of documentation, or missing feature, please file an issue on GitHub.

Please use Bunny's mailing list for questions, investigations, and discussions. GitHub issues should be used for specific, well understood, actionable maintainers and contributors can work on.

When filing an issue, please specify which Bunny and RabbitMQ versions you are using, provide recent RabbitMQ log file contents, full exception stack traces, and steps to reproduce (or failing test cases).

Other Ruby RabbitMQ Clients

The other widely used Ruby RabbitMQ client is March Hare (JRuby-only). It's a mature library that require RabbitMQ 3.3.x or later.

Contributing

See CONTRIBUTING.md for more information about running various test suites.

License

Released under the MIT license.