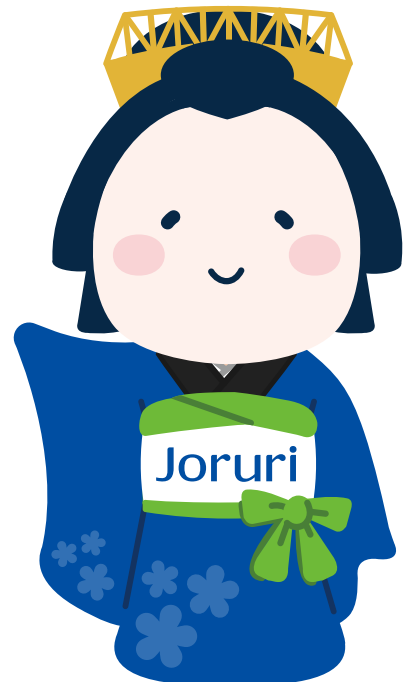


Joruri Mail 2022

JoruriMail2022 バージョンアップ手順書

2026-09-25 第三版
サイトブリッジ株式会社



1. 注意事項・事前準備
2. Rubyのアップグレード
3. サービスの停止
4. ソースコードの差し替え
5. 検索インデックスの再作成
6. サービスの再起動
7. ログインの確認

注意事項・事前準備

JoruriMail2022 Release3がインストールされている環境が対象の手順書です。
JoruriMail2022のバージョンアップを行う際に関連サービスの停止を行います。
そのため、バージョンアップ中はJoruriMail2022にアクセスできません。
また、本手順書は1台構成を想定しています。

事前準備として、Release4のソースコードをサーバーにアップロードします。
アップロード先パス: /usr/local/src/joruri-mail-2022-v4.0.0.tar.gz

Rubyのアップグレード

rbenvを更新します。

```
# cd /usr/local/rbenv/; git pull
# cd /usr/local/rbenv/plugins/rbenv-vars/; git pull
# cd /usr/local/rbenv/plugins/ruby-build/; git pull
```

rubyをインストールします。

```
# rbenv install 3.4.8
# rbenv global 3.4.8
# rbenv rehash
# ruby -v
```

インストールしたrubyのバージョンが表示されます。3.4.8と表示されれば成功です。

```
ruby 3.4.8 (2025-12-17 revision 995b59f666) +PRISM [x86_64-linux]
[root@localhost ~]#
```

MeCab-Rubyを再インストールします。

```
# cd /usr/local/src/mecab/mecab/ruby
# ruby extconf.rb && make && make install
# ldconfig
# ldconfig -p | grep "/usr/local/lib"
```

```
[root@localhost ruby]# ldconfig -p | grep "/usr/local/lib"
    libmecab.so.2 (libc6,x86-64) => /usr/local/lib/libmecab.so.2
    libmecab.so (libc6,x86-64) => /usr/local/lib/libmecab.so
```

Node.jsのアップグレード

nodenvを更新します。

```
# cd /usr/local/nodenv/; git pull
```

```
# cd /usr/local/nodenv/plugins/node-build/; git pull
```

nodejsをインストールします。

```
# nodenv install 24.12.0
```

```
# nodenv global 24.12.0
```

```
# nodenv rehash
```

```
# node -v
```

インストールしたnodejsのバージョンが表示されます。24.12.0と表示されれば成功です。

```
[root@localhost ~]# node -v
v24.12.0
```

pnpmをインストールします。

```
# cd
```

```
# npm install -g pnpm
```

```
# nodenv rehash
```

```
# pnpm -v
```

インストールしたpnpmのバージョンが表示されます。バージョン11.x以上が表示されれば成功です。

```
[root@localhost ~]# pnpm -v
12.3.4
```

Nginxのアップグレード

nginxを更新します。

```
# dnf -y install nginx
```

```
# nginx -v
```

インストールしたnginxのバージョンが表示されます。バージョン1.30.x以上が表示されれば成功です。

```
[root@localhost ~]# nginx -v
nginx version: nginx/1.30.4
```

Valkeyのインストール

valkeyをインストールします。

```
# dnf -y install valkey
```

```
# valkey-server -v
```

インストールしたvalkeyのバージョンが表示されます。バージョン8.0.x以上が表示されれば成功です。

```
[root@localhost ~]# valkey-server -v
Valkey server v=8.0.9 sha=00000000:1 malloc=jemalloc-5.3.0 bits=64 build=6913de585e6611e0
```

valkeyを設定します。

```
# vi /etc/valkey/valkey.conf
```

```
---
```

```
# 下記の行を追加します。環境毎に適切な値を設定してください。
```

```
maxmemory 500mb
```

```
maxmemory-policy allkeys-lru
```

```
---
```

サービスの停止

redisを停止します。

```
# systemctl stop redis && systemctl disable redis
# systemctl status redis
```

Active: inactive と表示されることを確認します。

nginxを停止します。

```
# systemctl stop nginx
# systemctl status nginx
```

Active: inactive と表示されることを確認します。

pumaを停止します。

```
# systemctl stop pwm_puma
# systemctl status pwm_puma
```

Active: inactive と表示されることを確認します。

delayed_jobを停止します。

```
# systemctl stop pwm_delayed_job
# systemctl status pwm_delayed_job
```

Active: inactive と表示されることを確認します。

crondを停止します。

```
# systemctl stop crond
# systemctl status crond
```

Active: inactive と表示されることを確認します。

ソースコード差し替え（1）

既存のJoruriMail2022ソースコードとRelease4のソースコードを差し替えます。

前回インストールしたpwmのソースコードが/usr/local/src配下に存在するか確認します。

```
# ls -lh /usr/local/src/pwm
```

存在する場合はリネームします。

```
# mv /usr/local/src/pwm /usr/local/src/pwm_v3
```

Release4のソースコードを解凍します。

```
# cd /usr/local/src
```

```
# tar -xvzf joruri-mail-2022-v4.0.0.tar.gz
```

既存のソースコードをリネームして退避します。

```
# mv /var/www/pwm /var/www/pwm_v3
```

Release4のソースコードを設置します。

```
# cp -r /usr/local/src/pwm /var/www/pwm
```

```
# chown -R pwm:pwm /var/www/pwm
```

pwmユーザーに変更します。

```
# su - pwm
```

ソースコードのパーミッションを変更します。※¥はバックスラッシュに置き換えてください。

```
$ find /var/www/pwm -type d -exec chmod 755 {} ¥;
```

既存のソースコードの設定ファイルをRelease4にコピーします。

```
$ cd /var/www/pwm
```

```
$ cp /var/www/pwm_v3/config/*.yml /var/www/pwm/config/.
```

```
$ cp /var/www/pwm_v3/config/credentials/*  
/var/www/pwm/config/credentials/.
```

ソースコード差し替え（2）

gemライブラリをインストールします。

```
$ cp config/samples/bundler/Gemfile.engines .  
$ bundle config build.pg --with-pg-config=/usr/pgsql-15/bin/pg_config  
$ bundle config set --local path 'vendor/bundle'  
$ bundle config set --local without 'development test'  
$ bundle install  
$ bundle list
```

DBを更新します。

```
$ ./bin/rails db:migrate RAILS_ENV=production
```

メールアプリケーション向けのGemfile.enginesをコピーします。

```
$ mv Gemfile.engines  
config/samples/bundler/Gemfile.engines.bk_yyyymmdd  
$ mv config/samples/bundler/Gemfile.mail.engines Gemfile.engines
```

※旧バージョンでコメントアウトしている箇所があれば同様にコメントアウトします。

```
$ vi Gemfile.engines
```

追加アプリケーションをインストールします。

```
$ bundle install  
$ bundle list
```

追加アプリケーション分のDB更新を反映します。

```
$ ./bin/rails db:migrate RAILS_ENV=production
```

jsライブラリを更新します。

```
$ rm -rf node_modules  
$ pnpm install --production  
$ pnpm list  
$ bin/install/assets.sh
```

デフォルト設定ファイルをコピーします。

```
$ cp config/original/core.yml config/core.yml
```

ソースコード差し替え（3）

アセットをコンパイルします。

```
$ ./bin/rails assets:precompile RAILS_ENV=production
```

最新のassetsファイルが作成されていることを確認します。

```
$ ls -l public/assets/**/*
```

cronに定期実行処理を追加します。

```
$ RAILS_ENV=production bundle exec whenever --update-crontab
```

railsコンソールを起動できることを確認します。

```
$ ./bin/rails console -e production
```

```
[pwm@localhost pwm]$ ./bin/rails console -e production
Loading production environment (Rails 8.1.3.1)
pwm(prod):001> █
```

エラーが出力されないことを確認したら、railsコンソールを終了します。

```
> exit
```

nginx設定ファイルを旧バージョンからコピーします。

```
$ cp -r ../pwm_v3/config/nginx/* config/nginx/
```

サービスの再起動

rootユーザーに切り替えます。

```
$ su -
```

valkeyを起動します。

```
# systemctl enable valkey && systemctl start valkey
```

```
# systemctl status valkey
```

Active: active (running)と表示されることを確認します。

pumaを起動します。

```
# systemctl start pwm_puma
```

```
# systemctl status pwm_puma
```

Active: active (running)と表示されることを確認します。

delayed_jobを起動します。

```
# systemctl start pwm_delayed_job
```

```
# systemctl status pwm_delayed_job
```

Active: active (running)と表示されることを確認します。

crondを起動します。

```
# systemctl start crond
```

```
# systemctl status crond
```

Active: active (running)と表示されることを確認します。

nginxを起動します。

```
# systemctl start nginx
```

```
# systemctl status nginx
```

Active: active (running)と表示されることを確認します。

メールの初期ラベルの作成

メールの初期ラベルを作成します。

```
# su - pwm
```

```
$ cd /var/www/pwm
```

```
$ ./bin/rails pwm_wmail:labels:create_defaults RAILS_ENV=production
```

JavaとTikaのバージョンアップ

tika-serverを停止します。

```
$ su -  
# systemctl stop tika  
# systemctl status tika
```

Javaをインストールします。

```
# dnf -y install java-25-openjdk java-25-openjdk-devel  
# java -version
```

```
[root@localhost ~]# java -version  
openjdk version "25.0.4.1" 2026-08-18 LTS  
OpenJDK Runtime Environment (Red_Hat-25.0.4.1.1-1) (build 25.0.4.1+1-LTS)  
OpenJDK 64-Bit Server VM (Red_Hat-25.0.4.1.1-1) (build 25.0.4.1+1-LTS, mixed mode, sharing)
```

古いバージョンが表示される場合、インストールしたバージョンに切り替えます。

```
# update-alternatives --config 'java'
```

```
[root@localhost ~]# update-alternatives --config 'java'  
There are 3 programs which provide 'java'.  
  
  Selection    Command  
-----  
*   1          java-17-openjdk.x86_64 (/usr/lib/jvm/java-17-openjdk-17.0.20.1.1-1.2.el9.alma.1.x86_64/bin/java)  
+   2          java-21-openjdk.x86_64 (/usr/lib/jvm/java-21-openjdk-21.0.12.1.1-1.2.el9.alma.1.x86_64/bin/java)  
    3          java-25-openjdk (/usr/lib/jvm/java-25-openjdk/bin/java)  
  
Enter to keep the current selection[+], or type selection number: 3
```

tika-serverをインストールします。

```
# curl https://archive.apache.org/dist/tika/3.3.0/tika-server-standard-3.3.0.jar -o /usr/local/sbin/tika-server.jar
```

tika-serverを起動します。

```
# systemctl start tika  
# systemctl status tika
```

確認します。

```
# curl http://localhost:9998/tika
```

```
[root@localhost ~]# curl http://localhost:9998/tika  
This is Tika Server (Apache Tika 3.3.0). Please PUT
```

Elasticsearchのアップグレード

Elasticsearchを8.9.1から9.3.4へアップグレードします。

まず、8.9.1から8.19.6へアップグレードします。

その後、8.19.6から9.3.4へアップグレードします。

※/etc/elasticsearch/elasticsearch.ymlのnetwork.hostのlocalhostを指定している場合の手順になります。

localhost以外を指定している場合は、localhostを指定している内容に置き換えてください。

必要なファイルをダウンロードします。

```
$ su -
```

```
# curl -fsSLO
```

```
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.19.6-x86_64.rpm
```

```
# curl -fsSLO
```

```
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.19.6-x86_64.rpm.sha512
```

```
# sha512sum -c elasticsearch-8.19.6-x86_64.rpm.sha512
```

```
-> elasticsearch-8.19.6-x86_64.rpm: 完了
```

```
# curl -fsSLO
```

```
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-9.3.4-x86_64.rpm
```

```
# curl -fsSLO
```

```
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-9.3.4-x86_64.rpm.sha512
```

```
# sha512sum -c elasticsearch-9.3.4-x86_64.rpm.sha512
```

```
→ elasticsearch-9.3.4-x86_64.rpm: 完了
```

Elasticsearch 8.9.1から8.19.6にアップグレード (1)

Elasticsearchを8.9.1から8.19.6へアップグレードします。

以下の手順を実施して、アップグレードします。

```
# curl -H "Content-Type: application/json" -XPUT
"localhost:9200/_cluster/settings?pretty" -d '{"persistent":
{"cluster.routing.allocation.enable": "primaries"}}'
```

```
# systemctl stop elasticsearch
```

```
# cd /usr/share/elasticsearch
# bin/elasticsearch-plugin remove analysis-icu
# bin/elasticsearch-plugin remove analysis-kuromoji
```

```
# cd
# rpm -Uvh elasticsearch-8.19.6-x86_64.rpm
```

```
# cd /usr/share/elasticsearch
# bin/elasticsearch-plugin install analysis-icu
# bin/elasticsearch-plugin install analysis-kuromoji
```

```
# systemctl daemon-reload
# systemctl start elasticsearch
# systemctl status elasticsearch
```

```
# curl -H "Content-Type: application/json" -XPUT
"localhost:9200/_cluster/settings?pretty" -d '{"persistent":
{"cluster.routing.allocation.enable": null}}'
```

Elasticsearch 8.9.1から8.19.6にアップグレード (2)

正常にアップグレードできたか確認します。

```
# curl -H "Content-Type: application/json" -XGET  
"localhost:9200/_cat/health?v"
```

"status"が"green"と表示されていれば正常です。

```
[pwm@localhost pwm]$ curl -H "Content-Type: application/json" -XGET "localhost:9200/_cat/health?v"  
epoch      timestamp cluster status node.total node.data shards pri relo init unassign pending_tasks max_task_wait_time active_shards_percent  
1788927534 04:18:54  pwm    green      1          1        0    0    0    0          0              -             100.0%
```

Elasticsearch 8.19.6から9.3.4にアップグレード (1)

Elasticsearchを8.19.6から9.3.4へアップグレードします。

以下の手順を実施して、アップグレードします。

```
# curl -H "Content-Type: application/json" -XPUT
"localhost:9200/_cluster/settings?pretty" -d '{"persistent":
{"cluster.routing.allocation.enable": "primaries"}}'
```

```
# systemctl stop elasticsearch
```

```
# cd /usr/share/elasticsearch
# bin/elasticsearch-plugin remove analysis-icu
# bin/elasticsearch-plugin remove analysis-kuromoji
```

```
# cd
# rpm -Uvh elasticsearch-9.3.4-x86_64.rpm
```

```
# cd /usr/share/elasticsearch
# bin/elasticsearch-plugin install analysis-icu
# bin/elasticsearch-plugin install analysis-kuromoji
```

```
# systemctl daemon-reload
# systemctl start elasticsearch
# systemctl status elasticsearch
```

```
# curl -H "Content-Type: application/json" -XPUT
"localhost:9200/_cluster/settings?pretty" -d '{"persistent":
{"cluster.routing.allocation.enable": null}}'
```

Elasticsearch 8.19.6から9.3.4にアップグレード (2)

正常にアップグレードできたか確認します。

```
# curl -H "Content-Type: application/json" -XGET  
"localhost:9200/_cat/health?v"
```

"status"が"green"と表示されていれば正常です。

```
[pwm@localhost pwm]$ curl -H "Content-Type: application/json" -XGET "localhost:9200/_cat/health?v"  
epoch      timestamp cluster status node.total node.data shards pri relo init unassign pending_tasks max_task_wait_time active_shards_percent  
1788927534 04:18:54  pwm    green      1          1        0    0    0    0          0              -             100.0%
```

Elasticsearch設定ファイルの変更

/etc/elasticsearch/elasticsearch.ymlのnode.rolesを変更します。

```
$ su -  
# vi /etc/elasticsearch/elasticsearch.yml  
※ingestとmlを削除します。
```

```
---  
node.roles: [data, master]  
---
```

Elasticsearchを再起動します。

```
# systemctl restart elasticsearch  
# systemctl status elasticsearch
```

ログインの確認

ブラウザでJoruriMail2022にアクセスしログインを確認します。

<https://pwm.localdomain.jp/>

※標準インストール時のURLです。インストール時の設定に合わせて適宜読み替えてください。

- ユーザーID: pwm
- パスワード: pwm

ログイン画面にVer4.0.0と表示されることを確認します。



The image shows the login screen for Joruri PWM Ver.4.0.0. The background is dark blue. At the top, the text 'Joruri PWM' is displayed in white and light green. Below it, 'PWM' and 'Ver.4.0.0' are shown in white. There are two white input fields: the first is labeled 'ユーザーID' (User ID) and the second is labeled 'パスワード' (Password). Below these fields is a blue button labeled 'ログイン' (Login). At the bottom right, there is a link labeled 'パスワード変更' (Change Password).