

Joruri Mail 2022

Joruri Mail 2022 バージョンアップ手順書

2025-08-22 第二版
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

Joruri Mail 2022 Release2がインストールされている環境が対象の手順書です。
Joruri Mail 2022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Joruri Mail 2022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、Release3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-v3.0.0.tar.gz`

Rubyのアップグレード（１）

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のアップグレード（2）

Rubyをインストールします。

```
# rbenv install 3.3.8
```

```
# rbenv global 3.3.8
```

```
# 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。3.3.4と表示されれば成功です。

```
[root@localhost ~]# ruby -v
ruby 3.3.4 (2024-07-09 revision be1089c8ec) [x86_64-linux]
```

MeCab-Rubyを再インストールします。

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

```
# cd mecab*
```

```
#ruby extconf.rb && make && make install
```

```
# mecab -v
```

インストールしたMeCabのバージョンが表示されます

```
[root@localhost mecab-0.996]# mecab -v
mecab of 0.996
```

サービスの停止

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Joruri Mail 2022ソースコードとRelease3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_v2
```

Release3のソースコードを解凍します。

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

```
# tar -xvzf joruri-mail-2022-v3.0.0.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_v2
```

Release3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。

```
$ 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Release2にコピーします。

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

```
$ cp /var/www/pwm_v2/config/*.yml /var/www/pwm/config/.
```

```
$ cp /var/www/pwm_v2/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
```

追加アプリケーションをインストールします。

```
$ bundle install
```

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。

```
$ yarn install --production  
$ yarn list  
$ ./bin/rails assets:precompile RAILS_ENV=production  
$ bin/install/assets.sh
```

設定ファイルを更新します。

```
$ cp config/original/storage.yml config/storage.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。

```
$ bundle exec rake 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。

```
$ 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 8.0.2)
pwm (prod) > 
```

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
```

ソースコード差し替え（４）

DBにプライマリテナントを追加します。

```
$ ./bin/rails pwm_core:tenants:create PRIMARY_TITLE="プライマリ"  
PRIMARY_HOSTNAME="pwm.localdomain.jp" RAILS_ENV  
=production  
---
```

※PRIMARY_TITLEとPRIMARY_HOSTNAMEはご利用の環境に合わせて設定してください。

プライマリテナントのファイル保存用ディレクトリを作成します。

```
$ mkdir -p fs/primary
```

uploadディレクトリをプライマリテナントのディレクトリに移動します。

```
$ mv upload fs/primary  
---
```

※uploadディレクトリがない場合は実行不要です。

storageディレクトリをプライマリテナントのディレクトリに移動します。

```
$ mv storage fs/primary  
---
```

※storageディレクトリがない場合は実行不要です。

ckeditorとace-buildsのパスを変更します。

```
$ mkdir public/assets/base  
$ cp -r public/assets/core/ckeditor public/assets/base/ckeditor  
$ ls -l public/assets/base/ckeditor  
$ cp -r public/assets/core/ace-builds public/assets/base/ace-builds  
$ ls -l public/assets/base/ace-builds
```

ソースコード差し替え（5）

nginxサンプル設定ファイルをコピーします。

```
$ cp -r config/samples/nginx/* config/nginx
```

```
$ vi config/nginx/pwm.d/primary.conf
```

```
---
```

```
server {
```

```
...
```

```
server_name pwm.localdomain.jp;
```

```
...
```

```
}
```

```
---
```

※環境に応じて適切にserver_nameを設定してください。

サービスの設定

rootユーザーに切り替えます。

```
$ su -
```

nginxを設定します。

```
# cp /var/www/pwm/config/samples/nginx/pwm_include.conf  
/etc/nginx/conf.d/pwm.conf
```

ファイルがコピーされていることを確認します。

```
# lh -lh /etc/nginx/conf.d/
```

```
[root@localhost ~]# ls -lh /etc/nginx/conf.d/  
合計 8.0K  
-rw-r--r--. 1 root root 1.1K 4月 23 21:58 default.conf.org  
-rw-r--r--. 1 root root 44 8月 21 10:04 pwm.conf
```

Elasticsearchの接続先ホストとポート番号の登録

現在の接続先ホストとポート番号を確認します。

```
# su - pwm
$ cd /var/www/pwm/
$ cat config/pwm_elasticclt.yml
```

```
elasticclt:
  elasticsearch:
    host: localhost
    port: 9200
```

設定ファイルを変更します。

```
$ vi config/pwm_elasticclt.yml
※第一階層を「elasticclt」から「nodes」に変更します。
※第二階層を「elasticsearch」から「localhost」に変更します。
```

```
nodes:
  localhost:
    host: localhost
    port: 9200
```

接続するノード名をテーブルに登録します。

```
$ ./bin/rails pwm_elasticclt:setting:change NODE_NAME=localhost
TENANT=primary RAILS_ENV=production
```

インデックス保存領域名の登録

現在のインデックス保存領域名を確認します。

```
# su - pwm
$ cd /var/www/pwm/
$ cat config/pwm_wmail_search.yml
---
wmail_search:
  elasticsearch:
    index_name: pwm_wmail_primary_production
---
```

index_nameの内容を控えておきます。

インデックス保存領域名をテーブルに登録します。INDEX_NAMEには、先ほど控えたindex_nameの内容を指定します。

```
$ ./bin/rails pwm_wmail_search:search:change_index_name
INDEX_NAME=pwm_wmail_primary_production TENANT=primary
RAILS_ENV=production
```

設定ファイルを削除します。

```
$ rm -f config/pwm_wmail_search.yml
```

サービスの再起動

rootユーザーに切り替えます。

```
$ su -
```

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。

ログインの確認

ブラウザでJoruri Mail 2022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Ver3.0.0と表示されることを確認します。



The image shows the login screen for Joruri PWM. The background is dark blue. At the top, the text 'Joruri PWM' is displayed in a large, white, sans-serif font, with 'PWM' in a light green color. Below this, the text 'PWM' and 'Ver.2.0.0' are shown in a smaller, white, sans-serif font. 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 with the text 'ログイン' (Login) in white. At the bottom right, there is a link labeled 'パスワード変更' (Change Password) in white.

検索インデックス再作成

全アカウントの検索インデックスを再作成します。

pwmユーザーに切り替えます。

```
# su - pwm
```

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

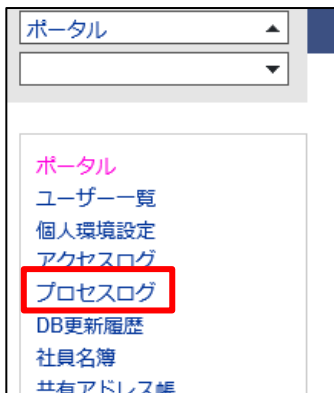
インデックスインポート(import_index)タスクを実行すると、プロセスログ画面から進行状況を確認できるので、完了まで待機します。

```
$ ./bin/rails pwm_wmail_search:search:create_index FORCE=true  
TENANT=primary RAILS_ENV=production
```

```
$ ./bin/rails pwm_wmail_search:search:set_refresh_interval INTERVAL=60s  
TENANT= primary RAILS_ENV=production
```

```
$ ./bin/rails pwm_wmail_search:search:import_index  
TARGET_ALL_ACCOUNT_CHECK=1 BATCH_SIZE=100 TENANT= primary  
RAILS_ENV=production
```

プロセスログ画面はシステム管理者でPWMにログインし、左上メニューを展開して「プロセスログ」を選択してください。



「インデックスインポート」プロセスのステータスを確認します。
ユーザー数やデータ規模に応じて複数回実行されます。
すべてが「終了」になるまで待機してください。

プロセスログ一覧								
キーワード		ステータス		検索		リセット		
ID	アプリケーション	プロセス	実行ユーザー	ステータス	終了日	実行時間	メモリ	
2391	メール	受信起動		終了	2024-10-16 15:10	0.10 s	183MB	
2390	メール	受信起動		終了	2024-10-16 15:09	0.11 s	182MB	
2389	メール	インデックスインポート		終了	2024-10-16 15:08	0.70 s	194MB	
2388	メール	受信起動		終了	2024-10-16 15:08	0.13 s	187MB	
2387	メール	スレッド解析起動		終了	2024-10-16 15:08	0.10 s	182MB	
2386	メール	受信起動		終了	2024-10-16 14:41	0.03 s	177MB	
2385	メール	受信起動		終了	2024-10-16 14:40	0.03 s	177MB	
2384	メール	受信起動		終了	2024-10-16 14:39	0.03 s	177MB	
2383	メール	受信起動		終了	2024-10-16 14:38	0.03 s	177MB	

プロセスログ画面を確認して、インデックスインポートが完了したら下記コマンドを実行します。

```
$ ./bin/rails pwm_wmail_search:search:set_refresh_interval INTERVAL=1s  
TENANT=primary RAILS_ENV=production
```