Connect-CMS V1→V2 グレードアップ

自動スクリプト（掲載用）

2026 年 3 月

> ℹ️ 赤字の「svXXXX」はサーバー番号です。ご自身の環境に合わせて置き換えてください。
> Xserver のサーバー番号はサーバーパネルの「サーバー情報」→「ホスト名」で確認できます。
> 例: sv12345.xserver.jp の場合、svXXXX → sv12345 に置き換えてください。

1. upgrade_to_v2.sh（V1→V2 バージョンアップ）

V1 から V2 へのバージョンアップをすべて自動化するメインスクリプトです。
バックアップ・git 切り替え・composer 更新・DB マイグレーションまで一括実行します。

```
#!/bin/bash
# ================================================================================
# Connect-CMS V1 → V2 バージョンアップスクリプト（Xserver 用）
# ================================================================================
#
# 【概要】
#   既存の Connect-CMS V1 環境を V2 にバージョンアップする自動スクリプト。
#   バックアップ → コード切り替え → composer → migrate → キャッシュクリア
#   の順で処理を行う。
#
# 【対応環境】
#   - サーバー     : Xserver（レンタルサーバー）
#   - 移行前       : Connect-CMS V1（PHP 7.4〜8.1 / Laravel 8）
#   - 移行後       : Connect-CMS V2（PHP 8.1〜8.3 / Laravel 10）
#
# 【重要な前提条件】
#   - PHP 8.1 以上が Xserver に存在すること（ls /usr/bin/php8* で確認）
#   - Xserver サーバーパネルの PHP Ver.切替で PHP 8.2 以上に変更済みであること
#     ※ SSH の PHP と Web サーバーの PHP は別管理のため両方の変更が必要
#   - DB バックアップ用の十分なディスク容量があること
#   - 作業前にサイトのアクセスが少ない時間帯を選ぶこと
#
# 【使い方】
#   1. FileZilla でホームディレクトリ（/home/USERNAME/）にアップロード
#   2. SSH 接続後、ホームディレクトリで以下を実行
#      $ chmod +x upgrade_to_v2.sh
#      $ ./upgrade_to_v2.sh
#
# 【実行ステップ】
#   STEP 0 : 事前確認（Web サーバー PHP 版数変更の確認を含む）
#   STEP 1 : インストールディレクトリの指定（.env から DB・メール情報を自動取得）
#   STEP 2 : PHP バージョン確認・切り替え（8.3→8.2→8.1 の優先順で自動選択）
#   STEP 3 : バックアップ（DB + connect-cms ディレクトリ）
#   STEP 4 : メンテナンスモード ON
#   STEP 5 : V2 ブランチに切り替え
#            ※ git stash が必要な理由：
#              V1 インストール後に composer install を実行すると
#              composer.json・composer.lock が git 管理上「ローカル変更あり」になる。
#              この状態で git checkout 2 を実行すると必ず失敗するため
#              事前に git stash で変更を退避する必要がある。
#            ※ git config が必要な理由：
```

```
#               git stash はコミットを作成するため git ユーザー設定がないと
#               「fatal: empty ident name not allowed」エラーが発生する。
#               git ユーザー設定がない場合は自動で設定する。
#   STEP 6 : Composer ライブラリ更新（Laravel 8→10 の依存ライブラリ更新）
#   STEP 7 : .env 更新
#            ※ sed の置換方式について：
#               「^KEY=.*」形式で置換する。
#               初期値チェック方式（MAIL_HOST=smtp.mailtrap.io との一致）では
#               V1 で設定済みの値がある場合に置換されず入力値が末尾に追記されて
#               「svXXXX.xserver.jpsvXXXX.xserver.jp」のように二重になる問題が
#               発生したため現在の値に関わらず正しく上書きできる形式を使用する。
#   STEP 8 : キャッシュクリア
#   STEP 9 : DB マイグレーション
#   STEP 10: Seeder 実行
#   STEP 11: メンテナンスモード OFF・動作確認
#
# 【失敗した場合の復元手順】
#   restore_connect_cms.sh を使用してください。
#
# 【DB 文字コード変換について】（utf8 → utf8mb4）
#   本スクリプトは DB 文字コードの自動変換を行いません。
#   理由：データが壊れるリスクがあるため手動対応を推奨します。
#   ※ utf8 のまま V2 に移行しても日本語テキストは正常に動作します。
#     絵文字（4 バイト文字）を使う場合のみ変換が必要です。
#
#   【変換前に必ずバックアップを取ること】
#     $ mysqldump -u DB ユーザー名 -p DB 名 > backup_before_utf8mb4.sql
#
#   【変換手順（SSH コマンド）】
#     # ① DB 全体の文字コードを変更
#     $ mysql -u DB ユーザー名 -p DB 名 -e \
#         "ALTER DATABASE DB 名 CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;"
#     # ② 全テーブルを一括変換する SQL を生成して実行
#     $ mysql -u DB ユーザー名 -p DB 名 -e \
#         "SELECT CONCAT('ALTER TABLE \`',TABLE_NAME,'\` CONVERT TO CHARACTER SET utf8mb4
COLLATE utf8mb4_general_ci;')
#          FROM information_schema.TABLES
#          WHERE TABLE_SCHEMA='DB 名' AND TABLE_TYPE='BASE TABLE';" \
#         --silent --skip-column-names | mysql -u DB ユーザー名 -p DB 名
#     # ③ 変換結果の確認
#     $ mysql -u DB ユーザー名 -p DB 名 -e \
#         "SELECT TABLE_NAME, TABLE_COLLATION FROM information_schema.TABLES
#          WHERE TABLE_SCHEMA='DB 名';"
#     # ④ キャッシュクリア
#     $ php artisan config:clear
#
#   【変換後に問題が起きた場合の復元】
#     $ mysql -u DB ユーザー名 -p -e "DROP DATABASE DB 名; CREATE DATABASE DB 名;"
#     $ mysql -u DB ユーザー名 -p DB 名 < backup_before_utf8mb4.sql
#
# 【トラブルシューティング】
#   - PHP バージョンエラー
#     → ls /usr/bin/php8* で利用可能なバイナリを確認
#     → Xserver サーバーパネルの PHP Ver.切替で PHP 8.2 以上に変更
#        （SSH の PHP と Web サーバーの PHP は別管理のため両方変更が必要）
#   - git stash エラー
#     → 本スクリプトでは git config を自動設定するため通常は発生しない
#   - git checkout 2 エラー
#     → 本スクリプトでは git stash を自動実行するため通常は発生しない
#   - composer install エラー
#     → php -d allow_url_fopen=1 composer.phar install --no-dev
#   - DB マイグレーションエラー
#     → storage/logs/laravel-*.log を確認
#   - メール設定が消える・二重になる
#     → grep "^MAIL_" .env で確認
#     → vi .env で正しい値に修正後 php artisan config:clear
#     → 本スクリプトではそのまま引き継ぐため通常は発生しない
#   - サイトが表示されない（Composer PHP 版数エラー）
#     → Xserver サーバーパネルの PHP Ver.切替で PHP 8.2 以上に変更
#   - サイトが表示されない（500 エラー）
#     → storage/logs/laravel-*.log を確認
#     → php artisan config:clear && php artisan cache:clear を実行
#
# 【関連ファイル】
#   - install_connect_cms_v1.sh : V1 新規インストールスクリプト
```

```
#   - install_connect_cms_v2.sh : V2 新規インストールスクリプト
#   - backup_connect_cms.sh     : バックアップスクリプト
#   - restore_connect_cms.sh    : 復元スクリプト
#
# 【更新履歴】
#   2026-03-28 初版作成
#   2026-03-28 git stash を STEP 5 に追加
#              （V1 インストール後の composer.json 変更による git checkout 失敗対策）
#   2026-03-28 git config 自動設定を追加
#              （git stash には git ユーザー設定が必要なため）
#   2026-03-28 .env の MAIL 設定置換方式を ^KEY=.* 形式に修正
#              （V1 設定済みの値が二重になる問題を修正）
#   2026-03-28 STEP 0 に Web サーバー PHP バージョン変更の確認を追加
#              （SSH の PHP と Web サーバーの PHP は別管理のため両方変更が必要）
#   2026-03-28 PHP バイナリ自動検出にドット区切り（php8.1 等）を追加
#              （Xserver のバイナリ名がドット区切りのため）
#   2026-03-28 STEP 1 で .env のメール設定を表示・引き継ぎ確認を追加
#
# ================================================================================

set -e

# --------------------------------------------------------------------------------
# カラー定義
# --------------------------------------------------------------------------------
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
BLUE='\033[0;34m'
CYAN='\033[0;36m'
BOLD='\033[1m'
NC='\033[0m'

info()    { echo -e "${BLUE}[INFO]${NC} $1"; }
success() { echo -e "${GREEN}[OK]${NC}   $1"; }
warn()    { echo -e "${YELLOW}[WARN]${NC} $1"; }
error()   { echo -e "${RED}[ERROR]${NC} $1"; exit 1; }
step()    { echo -e "\n${BOLD}${CYAN}========================================${NC}"; \
            echo -e "${BOLD}${CYAN}  $1${NC}"; \
            echo -e "${BOLD}${CYAN}========================================${NC}"; }

# --------------------------------------------------------------------------------
# バナー
# --------------------------------------------------------------------------------
clear
echo -e "${BOLD}${BLUE}"
echo "  ____                                _       ____ __  __ ____"
echo " / ___|___  _ __  _ __   ___  ___| |_     / ___|  \/  / ___|"
echo "| |   / _ \| '_ \| '_ \ / _ \/ __| __|   | |   | |\/| \___ \\"
echo "| |__| (_) | | | | | | |  __/ (__| |_    | |___| |  | |___) |"
echo " \____\___/|_| |_|_| |_|\___|\___|\__|    \____|_|  |_|____/"
echo -e "${NC}"
echo -e "${BOLD}  V1 → V2 バージョンアップスクリプト for Xserver${NC}"
echo -e "  ${CYAN}PHP 8.1 以上必須 / Laravel 8 → 10 / バックアップ自動作成${NC}"
echo ""

# --------------------------------------------------------------------------------
# STEP 0: 事前確認
# --------------------------------------------------------------------------------
step "STEP 0: 事前確認"
echo ""
warn "このスクリプトは V1 から V2 への移行を行います。必ず事前に確認してください："
echo ""
echo "  1. PHP 8.1 以上が Xserver に存在すること（ls /usr/bin/php8* で確認可能）"
echo "  2. Xserver サーバーパネルの PHP Ver.切替で PHP 8.2 以上に変更済みであること"
echo "     ※ SSH の PHP と Web サーバーの PHP は別管理のため両方の変更が必要です"
echo "     ※ 変更方法: サーバーパネル → PHP Ver.切替 → 対象ドメイン → 8.2 または 8.3 を選択"
echo "  3. ディスクに十分な空き容量があること（バックアップ用）"
echo "  4. できるだけアクセスが少ない時間帯に実行すること"
echo "  5. 本番環境の場合は事前にテスト環境で動作確認を行うこと"
echo ""
warn "移行失敗時は restore_connect_cms.sh でバックアップから復元してください。"
echo ""
```

```
read -p "上記を確認しました。バージョンアップを開始しますか？ [y/N]: " confirm
[[ "$confirm" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }


# ------------------------------------------------------------------------------
# STEP 1: インストールディレクトリの指定
# ------------------------------------------------------------------------------
step "STEP 1: インストールディレクトリの指定"
echo ""


USERNAME=$(whoami)
info "サーバー ID（USERNAME）を自動検出: ${BOLD}${USERNAME}${NC}"
echo ""


DEFAULT_INSTALL_DIR="${HOME}/${USERNAME}.xsrv.jp/connect-cms"
echo "例）"
echo "  メインドメインの場合 : ${HOME}/ドメイン名/connect-cms"
echo "  サブドメインの場合   : ${HOME}/親ドメイン名/connect-cms-サブドメイン名"
echo ""
read -p "V1 のインストールディレクトリ [${DEFAULT_INSTALL_DIR}]: " INPUT_INSTALL_DIR
INSTALL_DIR="${INPUT_INSTALL_DIR:-$DEFAULT_INSTALL_DIR}"


[[ -d "$INSTALL_DIR" ]] || error "指定されたディレクトリが見つかりません: ${INSTALL_DIR}"
[[ -f "${INSTALL_DIR}/.env" ]] || error ".env が見つかりません。Connect-CMS のインストールディレクトリを確認してください。"


# .env から DB 情報・メール情報を自動取得
DB_NAME=$(grep "^DB_DATABASE=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)
DB_USER=$(grep "^DB_USERNAME=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)
DB_PASS=$(grep "^DB_PASSWORD=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)


CURRENT_VERSION=$(cat "${INSTALL_DIR}/config/version.php" 2>/dev/null | grep -o "'[^']*'" |
head -1 | tr -d "'" || echo "不明")
info "インストール先  : ${BOLD}${INSTALL_DIR}${NC}"
info "現在のバージョン: ${BOLD}${CURRENT_VERSION}${NC}"
info "DB 名           : ${BOLD}${DB_NAME}${NC}"
echo ""


# 現在の .env メール設定を表示
info "現在の .env メール設定（V2 移行後もそのまま引き継ぎます）："
grep "^MAIL_" "${INSTALL_DIR}/.env" | while IFS= read -r line; do
    echo "    ${line}"
done
echo ""


read -p "この環境を V2 にアップグレードしますか？ [y/N]: " confirm2
[[ "$confirm2" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }


# ------------------------------------------------------------------------------
# STEP 2: PHP バージョン確認・切り替え
# ------------------------------------------------------------------------------
step "STEP 2: PHP バージョン確認・切り替え"
echo ""


info "利用可能な PHP 8.x バイナリを検索中..."
PHP_BINS=$(ls /usr/bin/php8* 2>/dev/null || true)
[[ -z "$PHP_BINS" ]] && error "PHP 8.x のバイナリが見つかりません。V2 は PHP 8.1 以上が必須です。"


echo ""
info "見つかった PHP 8.x バイナリ:"
for bin in $PHP_BINS; do
    echo "    $bin  ($($bin -r 'echo PHP_VERSION;' 2>/dev/null || echo '取得失敗'))"
done
echo ""


# PHP 8.3 → 8.2 → 8.1 の優先順で自動選択（ドット区切り・アンダースコア両対応）
PHP_BIN=""
for candidate in php8.3 php8.2 php8.1 php83 php82 php81; do
    if [[ -x "/usr/bin/${candidate}" ]]; then
        PHP_BIN="/usr/bin/${candidate}"
```

```
        break
    fi
done

if [[ -z "$PHP_BIN" ]]; then
    read -p "使用する PHP バイナリのフルパスを入力してください: " PHP_BIN
    [[ -x "$PHP_BIN" ]] || error "指定された PHP バイナリが見つかりません: ${PHP_BIN}"
fi

info "使用する PHP: ${BOLD}${PHP_BIN}${NC} ($(${PHP_BIN} -r 'echo PHP_VERSION;'))"
mkdir -p "$HOME/bin"
ln -sf "$PHP_BIN" "$HOME/bin/php"

if ! grep -q 'export PATH=$HOME/bin:$PATH' "$HOME/.bashrc" 2>/dev/null; then
    echo 'export PATH=$HOME/bin:$PATH' >> "$HOME/.bashrc"
fi
export PATH="$HOME/bin:$PATH"

PHP_VERSION=$(php -r 'echo PHP_MAJOR_VERSION.".".PHP_MINOR_VERSION;')
php -r 'if(PHP_MAJOR_VERSION < 8 || (PHP_MAJOR_VERSION == 8 && PHP_MINOR_VERSION < 1))
{ exit(1); }' \
    || error "PHP 8.1 以上が必要ですが ${PHP_VERSION} が検出されました"
success "PHP バージョン設定完了（${PHP_VERSION}）"

# ------------------------------------------------------------------------------
# STEP 3: バックアップ
# ------------------------------------------------------------------------------
step "STEP 3: バックアップ"
echo ""

BACKUP_DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="${HOME}/backup_${BACKUP_DATE}"
mkdir -p "$BACKUP_DIR"
info "バックアップ先: ${BOLD}${BACKUP_DIR}${NC}"
echo ""

info "DB バックアップ中..."
mysqldump -u "$DB_USER" -p"$DB_PASS" "$DB_NAME" > "${BACKUP_DIR}/${DB_NAME}.sql" \
    || error "DB バックアップ失敗。DB 設定を確認してください。"
DB_SIZE=$(du -sh "${BACKUP_DIR}/${DB_NAME}.sql" | cut -f1)
success "DB バックアップ完了: ${BACKUP_DIR}/${DB_NAME}.sql（${DB_SIZE}）"

info "connect-cms ディレクトリをバックアップ中（しばらく時間がかかります）..."
cp -r "$INSTALL_DIR" "${BACKUP_DIR}/connect-cms-backup"
DIR_SIZE=$(du -sh "${BACKUP_DIR}/connect-cms-backup" | cut -f1)
success "ディレクトリバックアップ完了: ${BACKUP_DIR}/connect-cms-backup（${DIR_SIZE}）"

echo ""
info "バックアップ完了。失敗した場合は restore_connect_cms.sh で復元してください。"
echo "    バックアップ先: ${BACKUP_DIR}"
echo ""

# ------------------------------------------------------------------------------
# STEP 4: メンテナンスモード ON
# ------------------------------------------------------------------------------
step "STEP 4: メンテナンスモード ON"
echo ""

cd "$INSTALL_DIR"
php artisan down
success "メンテナンスモード ON（サイトへのアクセスを一時停止）"

# エラー時にメンテナンスモードを自動解除するトラップ
trap 'echo -e "\n${RED}[ERROR]${NC} エラーが発生しました。メンテナンスモードを解除します..."; php
artisan up 2>/dev/null; echo "メンテナンスモードを解除しました。restore_connect_cms.sh でバックアッ
プから復元してください。"' ERR

# ------------------------------------------------------------------------------
# STEP 5: V2 ブランチに切り替え
```

```
# ------------------------------------------------------------------------------
step "STEP 5: V2 ブランチに切り替え"
echo ""

# git stash は git ユーザー設定が必要なため事前に確認・自動設定する
# V1 インストール後に composer install を実行すると composer.json・composer.lock が
# git 管理上「ローカル変更あり」になるため、git checkout 2 の前に必ず git stash が必要。
# git stash はコミットを作成するため git ユーザー設定がないと
# 「fatal: empty ident name not allowed」エラーが発生する。
if [[ -z "$(git config --global user.email 2>/dev/null)" ]]; then
    info "git ユーザー設定がないため自動設定します..."
    git config --global user.email "${USERNAME}@svXXXX.xserver.jp"
    git config --global user.name "${USERNAME}"
    success "git ユーザー設定完了"
fi

info "ローカルの変更を一時退避中（git stash）..."
git stash
success "git stash 完了（composer.json 等のローカル変更を退避）"

info "master ブランチに切り替え中..."
git checkout master
git pull
success "master 更新完了"

info "V2 ブランチ「2」に切り替え中..."
git checkout 2
LATEST_TAG=$(git describe --tags --abbrev=0)
git checkout "$LATEST_TAG"
success "V2 最新安定版 ${BOLD}${LATEST_TAG}${NC} に切り替え完了"

# ------------------------------------------------------------------------------
# STEP 6: Composer ライブラリ更新
# ------------------------------------------------------------------------------
step "STEP 6: Composer ライブラリ更新"
echo ""

info "依存ライブラリを更新中（数分かかります）..."
php -d allow_url_fopen=1 composer.phar install --no-dev --no-interaction
success "Composer ライブラリ更新完了"

# ------------------------------------------------------------------------------
# STEP 7: .env 更新
# ------------------------------------------------------------------------------
step "STEP 7: .env 更新"
echo ""

# MAIL_DRIVER → MAIL_MAILER への変更
if grep -q "^MAIL_DRIVER=" .env; then
    sed -i 's/^MAIL_DRIVER=/MAIL_MAILER=/' .env
    success ".env: MAIL_DRIVER → MAIL_MAILER に変更"
else
    info ".env: MAIL_MAILER はすでに設定済みのためスキップ"
fi

# SESSION_SECURE_COOKIE の追加・有効化
if grep -q "^#SESSION_SECURE_COOKIE=true" .env; then
    sed -i 's/^#SESSION_SECURE_COOKIE=true/SESSION_SECURE_COOKIE=true/' .env
    success ".env: SESSION_SECURE_COOKIE=true を有効化"
elif ! grep -q "SESSION_SECURE_COOKIE" .env; then
    echo -e "\nSESSION_SECURE_COOKIE=true" >> .env
    success ".env: SESSION_SECURE_COOKIE=true を追加"
else
    info ".env: SESSION_SECURE_COOKIE はすでに設定済みのためスキップ"
fi

# APP_URL が http:// のままの場合は https:// に変更
CURRENT_URL=$(grep "^APP_URL=" .env | cut -d'=' -f2)
if [[ "$CURRENT_URL" == http://* ]]; then
    NEW_URL="${CURRENT_URL/http:\/\//https:\/\/}"
```

```
    sed -i "s|^APP_URL=.*|APP_URL=${NEW_URL}|" .env
    warn ".env: APP_URL を http:// → https:// に変更しました（${NEW_URL}）"
else
    info ".env: APP_URL はすでに https:// のためスキップ"
fi

# メール設定はそのまま引き継ぐ（変更しない）
# ※ 過去に初期値チェック方式で置換を試みたところ、V1 で設定済みの値が
#   末尾に追記されて「svXXXX.xserver.jpsvXXXX.xserver.jp」のように
#   二重になる問題が発生したため、メール設定は一切変更しない方針とした。
info ".env: メール設定はそのまま引き継ぎます（変更なし）"

echo ""
info "更新後の.env メール設定："
grep "^MAIL_" .env | while IFS= read -r line; do
    echo "    ${line}"
done
success ".env 更新完了"

# ------------------------------------------------------------------------------
# STEP 8: キャッシュクリア
# ------------------------------------------------------------------------------
step "STEP 8: キャッシュクリア"
echo ""

php artisan config:clear
php artisan cache:clear
php artisan view:clear
success "キャッシュクリア完了"

# ------------------------------------------------------------------------------
# STEP 9: DB マイグレーション
# ------------------------------------------------------------------------------
step "STEP 9: DB マイグレーション"
echo ""

warn "V1→V2 の大量のマイグレーションが実行されます。時間がかかる場合があります。"
echo ""
php artisan migrate --force || error "DB マイグレーション失敗。ログを確認: storage/logs/laravel-
*.log"
success "DB マイグレーション完了"

# ------------------------------------------------------------------------------
# STEP 10: Seeder 実行
# ------------------------------------------------------------------------------
step "STEP 10: Seeder 実行"
echo ""

php artisan db:seed --force
success "Seeder 完了"

# ------------------------------------------------------------------------------
# STEP 11: メンテナンスモード OFF
# ------------------------------------------------------------------------------
step "STEP 11: メンテナンスモード OFF"
echo ""

trap - ERR
php artisan up
success "メンテナンスモード OFF（サイトへのアクセスを再開）"

# ------------------------------------------------------------------------------
# 完了
# ------------------------------------------------------------------------------
echo ""
echo -e "${BOLD}${GREEN}"
echo "  ================================================"
echo "   V1 → V2 バージョンアップ完了！"
echo "  ================================================"
echo -e "${NC}"
```

```
echo -e "  バージョン    : ${BOLD}${LATEST_TAG}${NC}"
echo -e "  インストール先: ${BOLD}${INSTALL_DIR}${NC}"
echo -e "  バックアップ  : ${BOLD}${BACKUP_DIR}${NC}"
echo ""
echo -e "${YELLOW}  【確認事項】${NC}"
echo "  1. ブラウザでサイトにアクセスして表示を確認してください"
echo "  2. 管理画面にログインして各機能の動作を確認してください"
echo "  3. 管理画面のメール送信テストでメールが送れるか確認してください"
echo "  4. 問題なければバックアップディレクトリは削除して構いません"
echo "     $ rm -rf ${BACKUP_DIR}"
echo ""
echo -e "${CYAN}  公式サイト: https://connect-cms.jp${NC}"
echo ""
```

2. backup_connect_cms.sh（バックアップ）

Connect-CMS の DB とディレクトリを日時付きフォルダにバックアップします。
個人情報の置き換えは不要です。

```
#!/bin/bash
# ============================================================================
# Connect-CMS バックアップスクリプト（Xserver 用）
# ============================================================================
#
#  【概要】
#   Connect-CMS のデータをバックアップするスクリプト。
#   DB（mysqldump）と connect-cms ディレクトリを日時付きフォルダに保存する。
#   新規インストール・バージョンアップ・定期バックアップなど用途を問わず使用可能。
#
#  【バックアップ対象】
#   1. DB（mysqldump による SQL ファイル）
#   2. connect-cms ディレクトリ全体（.env・カスタムテーマ含む）
#
#  【バックアップ先】
#   ~/backup_YYYYMMDD_HHMMSS/
#     ├── DB名.sql               ← DB バックアップ
#     └── connect-cms-backup/    ← ディレクトリバックアップ
#
#  【使い方】
#   1. FileZilla でホームディレクトリ（/home/USERNAME/）にアップロード
#   2. SSH 接続後、ホームディレクトリで以下を実行
#      $ chmod +x backup_connect_cms.sh
#      $ ./backup_connect_cms.sh
#
#  【復元方法】
#   restore_connect_cms.sh を使用してください。
#
#  【注意事項】
#   - バックアップにはディスク容量が必要です（アップロードファイルが多い場合は特に）
#   - 古いバックアップは定期的に削除してください
#   - バックアップ後は ~/backup_YYYYMMDD_HHMMSS/ を安全な場所にコピーすることを推奨
#
#  【関連ファイル】
#   - restore_connect_cms.sh    : 復元スクリプト
#   - install_connect_cms_v1.sh : V1 新規インストールスクリプト
#   - install_connect_cms_v2.sh : V2 新規インストールスクリプト
#   - upgrade_to_v2.sh          : V1→V2 バージョンアップスクリプト
#
#  【更新履歴】
#   2026-03-28 初版作成
#
# ============================================================================

set -e

# ----------------------------------------------------------------------------
# カラー定義
# ----------------------------------------------------------------------------
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
BLUE='\033[0;34m'
CYAN='\033[0;36m'
BOLD='\033[1m'
NC='\033[0m'

info()    { echo -e "${BLUE}[INFO]${NC} $1"; }
success() { echo -e "${GREEN}[OK]${NC}   $1"; }
warn()    { echo -e "${YELLOW}[WARN]${NC} $1"; }
error()   { echo -e "${RED}[ERROR]${NC} $1"; exit 1; }
step()    { echo -e "\n${BOLD}${CYAN}========================================${NC}"; \
            echo -e "${BOLD}${CYAN}  $1${NC}"; \
            echo -e "${BOLD}${CYAN}========================================${NC}"; }
```

```
# ------------------------------------------------------------------------------
# バナー
# ------------------------------------------------------------------------------
clear
echo -e "${BOLD}${BLUE}"
echo "  ____                                _          ____ __  __ ____"
echo " / ___|___  _ __  _ __    ___   ___| |_      / ___|  \/  / ___|"
echo "| |   / _ \| '_ \| '_ \ / _ \ / __| __|    | |   | |\/| \___ \\"
echo "| |__| (_) | | | | | | |  __/ (__| |_     | |___| |  | |___) |"
echo " \____\___/|_| |_|_| |_|\___|\___|\__|     \____|_|  |_|____/"
echo -e "${NC}"
echo -e "${BOLD}  バックアップスクリプト for Xserver${NC}"
echo -e "  ${CYAN}DB + connect-cms ディレクトリを日時付きフォルダに保存${NC}"
echo ""

# ------------------------------------------------------------------------------
# STEP 0: 事前確認
# ------------------------------------------------------------------------------
step "STEP 0: 事前確認"
echo ""
warn "バックアップにはディスク容量が必要です。"
warn "アップロードファイルが多い場合は特に容量を確認してください。"
echo ""
read -p "バックアップを開始しますか？ [y/N]: " confirm
[[ "$confirm" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }

# ------------------------------------------------------------------------------
# STEP 1: 設定入力
# ------------------------------------------------------------------------------
step "STEP 1: 設定入力"
echo ""

USERNAME=$(whoami)
info "サーバー ID（USERNAME）を自動検出: ${BOLD}${USERNAME}${NC}"
echo ""

# インストールディレクトリ
DEFAULT_INSTALL_DIR="${HOME}/${USERNAME}.xsrv.jp/connect-cms"
read -p "connect-cms のインストールディレクトリ [${DEFAULT_INSTALL_DIR}]: " INPUT_INSTALL_DIR
INSTALL_DIR="${INPUT_INSTALL_DIR:-$DEFAULT_INSTALL_DIR}"
[[ -d "$INSTALL_DIR" ]] || error "ディレクトリが見つかりません: ${INSTALL_DIR}"
[[ -f "${INSTALL_DIR}/.env" ]] || error ".env が見つかりません。インストールディレクトリを確認してください。"

# .env から DB 情報を自動取得
DB_NAME=$(grep "^DB_DATABASE=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)
DB_USER=$(grep "^DB_USERNAME=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)
DB_PASS=$(grep "^DB_PASSWORD=" "${INSTALL_DIR}/.env" | cut -d'=' -f2)

info "インストールディレクトリ: ${BOLD}${INSTALL_DIR}${NC}"
info "DB 名（.env から自動取得）  : ${BOLD}${DB_NAME}${NC}"
info "DB ユーザー                 : ${BOLD}${DB_USER}${NC}"
echo ""

read -p "この設定でバックアップしますか？ [y/N]: " confirm2
[[ "$confirm2" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }

# ------------------------------------------------------------------------------
# STEP 2: バックアップ先の作成
# ------------------------------------------------------------------------------
step "STEP 2: バックアップ先の作成"
echo ""

BACKUP_DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="${HOME}/backup_${BACKUP_DATE}"
mkdir -p "$BACKUP_DIR"
info "バックアップ先: ${BOLD}${BACKUP_DIR}${NC}"

# ------------------------------------------------------------------------------
```

```
# STEP 3: DBバックアップ
# ------------------------------------------------------------------------------
step "STEP 3: DBバックアップ"
echo ""

info "DBをバックアップ中..."
mysqldump -u "$DB_USER" -p"$DB_PASS" "$DB_NAME" > "${BACKUP_DIR}/${DB_NAME}.sql" \
    || error "DBバックアップ失敗。DB設定を確認してください。"

DB_SIZE=$(du -sh "${BACKUP_DIR}/${DB_NAME}.sql" | cut -f1)
success "DBバックアップ完了: ${BACKUP_DIR}/${DB_NAME}.sql（${DB_SIZE}）"


# ------------------------------------------------------------------------------
# STEP 4: ディレクトリバックアップ
# ------------------------------------------------------------------------------
step "STEP 4: connect-cmsディレクトリバックアップ"
echo ""

info "ディレクトリをバックアップ中（しばらく時間がかかります）..."
cp -r "$INSTALL_DIR" "${BACKUP_DIR}/connect-cms-backup"

DIR_SIZE=$(du -sh "${BACKUP_DIR}/connect-cms-backup" | cut -f1)
success "ディレクトリバックアップ完了: ${BACKUP_DIR}/connect-cms-backup（${DIR_SIZE}）"


# ------------------------------------------------------------------------------
# 完了
# ------------------------------------------------------------------------------
TOTAL_SIZE=$(du -sh "$BACKUP_DIR" | cut -f1)
echo ""
echo -e "${BOLD}${GREEN}"
echo "  ================================================"
echo "   バックアップ完了！"
echo "  ================================================"
echo -e "${NC}"
echo -e "  バックアップ先   : ${BOLD}${BACKUP_DIR}${NC}"
echo -e "  DB               : ${BOLD}${BACKUP_DIR}/${DB_NAME}.sql${NC}"
echo -e "  ディレクトリ     : ${BOLD}${BACKUP_DIR}/connect-cms-backup${NC}"
echo -e "  合計サイズ       : ${BOLD}${TOTAL_SIZE}${NC}"
echo ""
echo -e "${YELLOW}  【推奨】バックアップフォルダをローカルPCにダウンロードして保管してください${NC}"
echo -e "  FileZillaで ${BACKUP_DIR} をダウンロードできます"
echo ""
echo -e "${CYAN}  復元する場合は restore_connect_cms.sh を使用してください${NC}"
echo ""
```

3. restore_connect_cms.sh（復元）

バックアップから Connect-CMS を復元します。

個人情報の置き換えは不要です。

```
#!/bin/bash
# ============================================================================
# Connect-CMS 復元スクリプト（Xserver 用）
# ============================================================================
#
# 【概要】
#   backup_connect_cms.sh で作成したバックアップから Connect-CMS を復元する
#   スクリプト。DB の復元とディレクトリの復元を行う。
#
# 【復元対象】
#   1. DB（SQL ファイルからの復元）
#   2. connect-cms ディレクトリ全体
#
# 【前提条件】
#   - backup_connect_cms.sh で作成したバックアップフォルダが存在すること
#   - バックアップフォルダの構成:
#       ~/backup_YYYYMMDD_HHMMSS/
#         ├── DB名.sql
#         └── connect-cms-backup/
#
# 【使い方】
#   1. FileZilla でホームディレクトリ（/home/USERNAME/）にアップロード
#   2. SSH 接続後、ホームディレクトリで以下を実行
#      $ chmod +x restore_connect_cms.sh
#      $ ./restore_connect_cms.sh
#
# 【注意事項】
#   - 復元すると現在の DB とディレクトリが上書きされます
#   - 復元前に現在の状態を別途バックアップすることを推奨します
#   - 復元中はサイトにアクセスできなくなります
#
# 【関連ファイル】
#   - backup_connect_cms.sh      : バックアップスクリプト
#   - install_connect_cms_v1.sh : V1 新規インストールスクリプト
#   - install_connect_cms_v2.sh : V2 新規インストールスクリプト
#   - upgrade_to_v2.sh           : V1→V2 バージョンアップスクリプト
#
# 【更新履歴】
#   2026-03-28 初版作成
#
# ============================================================================

set -e

# ----------------------------------------------------------------------------
# カラー定義
# ----------------------------------------------------------------------------
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
BLUE='\033[0;34m'
CYAN='\033[0;36m'
BOLD='\033[1m'
NC='\033[0m'

info()    { echo -e "${BLUE}[INFO]${NC} $1"; }
success() { echo -e "${GREEN}[OK]${NC}   $1"; }
warn()    { echo -e "${YELLOW}[WARN]${NC} $1"; }
error()   { echo -e "${RED}[ERROR]${NC} $1"; exit 1; }
step()    { echo -e "\n${BOLD}${CYAN}========================================${NC}"; \
            echo -e "${BOLD}${CYAN}  $1${NC}"; \
            echo -e "${BOLD}${CYAN}========================================${NC}"; }

# ----------------------------------------------------------------------------
```

```
# バナー
# ------------------------------------------------------------------------------
clear
echo -e "${BOLD}${BLUE}"
echo "  ____                                  _          ____ __  __ ____"
echo " / ___|___  _ __  _ __   ___  ___| |_     / ___|  \/  / ___|"
echo "| |   / _ \| '_ \| '_ \ / _ \/ __| __|   | |   | |\/| \___ \\"
echo "| |__| (_) | | | | | | |  __/ (__| |_    | |___| |  | |___) |"
echo " \____\___/|_| |_|_| |_|\___|\___|\__|    \____|_|  |_|____/"
echo -e "${NC}"
echo -e "${BOLD}  復元スクリプト for Xserver${NC}"
echo -e "  ${CYAN}バックアップから DB + connect-cms ディレクトリを復元${NC}"
echo ""


# ------------------------------------------------------------------------------
# STEP 0: 事前確認
# ------------------------------------------------------------------------------
step "STEP 0: 事前確認"
echo ""
warn "復元すると現在の DB とディレクトリが上書きされます。"
warn "現在の状態を残したい場合は先に backup_connect_cms.sh を実行してください。"
echo ""
read -p "復元を開始しますか？ [y/N]: " confirm
[[ "$confirm" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }


# ------------------------------------------------------------------------------
# STEP 1: バックアップフォルダの選択
# ------------------------------------------------------------------------------
step "STEP 1: バックアップフォルダの選択"
echo ""

# 利用可能なバックアップフォルダを一覧表示
info "利用可能なバックアップフォルダ:"
BACKUP_DIRS=$(ls -d ${HOME}/backup_* 2>/dev/null | sort -r || true)

if [[ -z "$BACKUP_DIRS" ]]; then
    error "バックアップフォルダが見つかりません。backup_connect_cms.sh でバックアップを作成してくださ
い。"
fi

i=1
declare -a BACKUP_LIST
while IFS= read -r dir; do
    SIZE=$(du -sh "$dir" 2>/dev/null | cut -f1)
    DATE_STR=$(basename "$dir" | sed 's/backup_//' | sed 's/_/ /')
    echo "  ${i}) $(basename $dir)   (${SIZE}) "
    BACKUP_LIST[$i]="$dir"
    ((i++))
done <<< "$BACKUP_DIRS"

echo ""
read -p "復元するバックアップ番号を入力してください [1]: " INPUT_NUM
SELECTED_NUM="${INPUT_NUM:-1}"
BACKUP_DIR="${BACKUP_LIST[$SELECTED_NUM]}"
[[ -d "$BACKUP_DIR" ]] || error "選択されたバックアップフォルダが見つかりません"

info "選択されたバックアップ: ${BOLD}${BACKUP_DIR}${NC}"

# バックアップフォルダの内容確認
SQL_FILES=$(ls "${BACKUP_DIR}"/*.sql 2>/dev/null || true)
[[ -z "$SQL_FILES" ]] && error "SQL ファイルが見つかりません: ${BACKUP_DIR}/*.sql"
[[ -d "${BACKUP_DIR}/connect-cms-backup" ]] || error "connect-cms-backup が見つかりません: $
{BACKUP_DIR}/connect-cms-backup"

# SQL ファイルから DB 名を取得
SQL_FILE=$(ls "${BACKUP_DIR}"/*.sql | head -1)
DB_NAME=$(basename "$SQL_FILE" .sql)
info "DB ファイル    : ${BOLD}${SQL_FILE}${NC}"
info "DB 名          : ${BOLD}${DB_NAME}${NC}"
info "ディレクトリ   : ${BOLD}${BACKUP_DIR}/connect-cms-backup${NC}"
echo ""
```

```
# ------------------------------------------------------------------------------
# STEP 2: 復元先の確認
# ------------------------------------------------------------------------------
step "STEP 2: 復元先の確認"
echo ""

# .env から DB 情報を取得（バックアップの connect-cms-backup から）
ENV_FILE="${BACKUP_DIR}/connect-cms-backup/.env"
[[ -f "$ENV_FILE" ]] || error ".env が見つかりません: ${ENV_FILE}"

DB_USER=$(grep "^DB_USERNAME=" "$ENV_FILE" | cut -d'=' -f2)
DB_PASS=$(grep "^DB_PASSWORD=" "$ENV_FILE" | cut -d'=' -f2)
INSTALL_DIR=$(grep "^APP_URL=" "$ENV_FILE" | cut -d'=' -f2 | sed 's|https\?://||')

# 復元先ディレクトリを確認
DEFAULT_INSTALL_DIR="${HOME}/$(echo $INSTALL_DIR | cut -d'/' -f1)/connect-cms"
read -p "復元先ディレクトリ [${DEFAULT_INSTALL_DIR}]: " INPUT_RESTORE_DIR
RESTORE_DIR="${INPUT_RESTORE_DIR:-$DEFAULT_INSTALL_DIR}"

echo ""
echo -e "  バックアップ元   : ${BOLD}${BACKUP_DIR}${NC}"
echo -e "  DB 名            : ${BOLD}${DB_NAME}${NC}"
echo -e "  DB ユーザー      : ${BOLD}${DB_USER}${NC}"
echo -e "  復元先ディレクトリ: ${BOLD}${RESTORE_DIR}${NC}"
echo ""
warn "上記の内容で復元します。現在の DB とディレクトリは上書きされます。"
echo ""
read -p "復元を実行しますか？ [y/N]: " confirm2
[[ "$confirm2" =~ ^[Yy]$ ]] || { echo "中止しました。"; exit 0; }

# ------------------------------------------------------------------------------
# STEP 3: DB 復元
# ------------------------------------------------------------------------------
step "STEP 3: DB 復元"
echo ""

info "DB を復元中..."
mysql -u "$DB_USER" -p"$DB_PASS" -e \
    "DROP DATABASE IF EXISTS \`${DB_NAME}\`; CREATE DATABASE \`${DB_NAME}\`;" \
    || error "DB 削除・再作成に失敗しました"

mysql -u "$DB_USER" -p"$DB_PASS" "$DB_NAME" < "$SQL_FILE" \
    || error "DB 復元に失敗しました"

success "DB 復元完了: ${DB_NAME}"

# ------------------------------------------------------------------------------
# STEP 4: ディレクトリ復元
# ------------------------------------------------------------------------------
step "STEP 4: connect-cms ディレクトリ復元"
echo ""

# 既存ディレクトリを退避
if [[ -d "$RESTORE_DIR" ]]; then
    FAILED_DIR="${RESTORE_DIR}_failed_$(date +%Y%m%d_%H%M%S)"
    info "既存ディレクトリを退避中: ${FAILED_DIR}"
    mv "$RESTORE_DIR" "$FAILED_DIR"
    success "退避完了: ${FAILED_DIR}"
fi

info "ディレクトリを復元中（しばらく時間がかかります）..."
cp -r "${BACKUP_DIR}/connect-cms-backup" "$RESTORE_DIR"
success "ディレクトリ復元完了: ${RESTORE_DIR}"

# ------------------------------------------------------------------------------
# STEP 5: キャッシュクリア
# ------------------------------------------------------------------------------
step "STEP 5: キャッシュクリア"
```

```
echo ""

cd "$RESTORE_DIR"

# PHP パスの設定
export PATH="$HOME/bin:$PATH"

php artisan config:clear 2>/dev/null && success "config キャッシュクリア完了" || warn "config
キャッシュクリア失敗（手動で実行してください）"
php artisan cache:clear  2>/dev/null && success "cache キャッシュクリア完了"  || warn "cache
キャッシュクリア失敗（手動で実行してください）"
php artisan view:clear   2>/dev/null && success "view キャッシュクリア完了"   || warn "view
キャッシュクリア失敗（手動で実行してください）"

# ------------------------------------------------------------------------------
# 完了
# ------------------------------------------------------------------------------
echo ""
echo -e "${BOLD}${GREEN}"
echo "  ================================================"
echo "   復元完了！"
echo "  ================================================"
echo -e "${NC}"
echo -e "  復元元バックアップ: ${BOLD}${BACKUP_DIR}${NC}"
echo -e "  復元先             : ${BOLD}${RESTORE_DIR}${NC}"
echo ""
echo -e "${YELLOW}  【確認事項】${NC}"
echo "  1. ブラウザでサイトにアクセスして表示を確認してください"
echo "  2. 管理画面にログインして各機能の動作を確認してください"
echo ""
if [[ -d "$FAILED_DIR" ]]; then
    echo -e "${CYAN}  退避した旧ディレクトリ: ${FAILED_DIR}${NC}"
    echo "  問題なければ削除して構いません："
    echo "  $ rm -rf ${FAILED_DIR}"
fi
echo ""
```