

WordPressのバージョンアップを安全に行うための ローカル検証環境活用術

本番サイトを壊さない 検証&メンテナンス手法

🎯 今日話す内容



1. 運用方法によるリスクの理解

運用方法によって生じる危険性と脆弱性リスクについて



2. ローカルツールの選択

検証に最適なツールの選び方について



3. バックアップの重要性

万が一に備える「復元ポイント」の作成の重要性について



4. 正しいアップデート手順

サイト障害を起こしにくい、安全な「アップデートの順番」について

なぜアップデートが必要か

「動いているなら触るな」の危険

システム運用では「正常に動いているシステムには手を加えない」という考え方をする担当者も存在しました。

しかし、脆弱性が日々発見される現代において、「**放置すること**」自体が最大のセキュリティリスクへと変化しています。

意識の変化と顧客からの相談

近年では、企業のWeb担当者やクライアントからのセキュリティ意識も高まっています。

「自社のサイトは放置されていて大丈夫？」という不安に対し**安全で安心なアップデート体制**が求められています。

✧ 現代のWebの脅威

攻撃は「自動化・AI化」されている

現在のサイバー攻撃は、ハッカーが特定のサイトを手作業で狙い撃ちしているわけではありません。

スキャンボット（自動巡回プログラム）が24時間体制でネット上のWordPressサイトを一斉にチェックしています。

注意：アクセス数の少ないマイナーサイトや小さな企業のブログでも、古いバージョンであれば機械的に一瞬で乗っ取られます。



🔧 検証環境の準備

本番環境でいきなり「更新」ボタンを押していませんか？

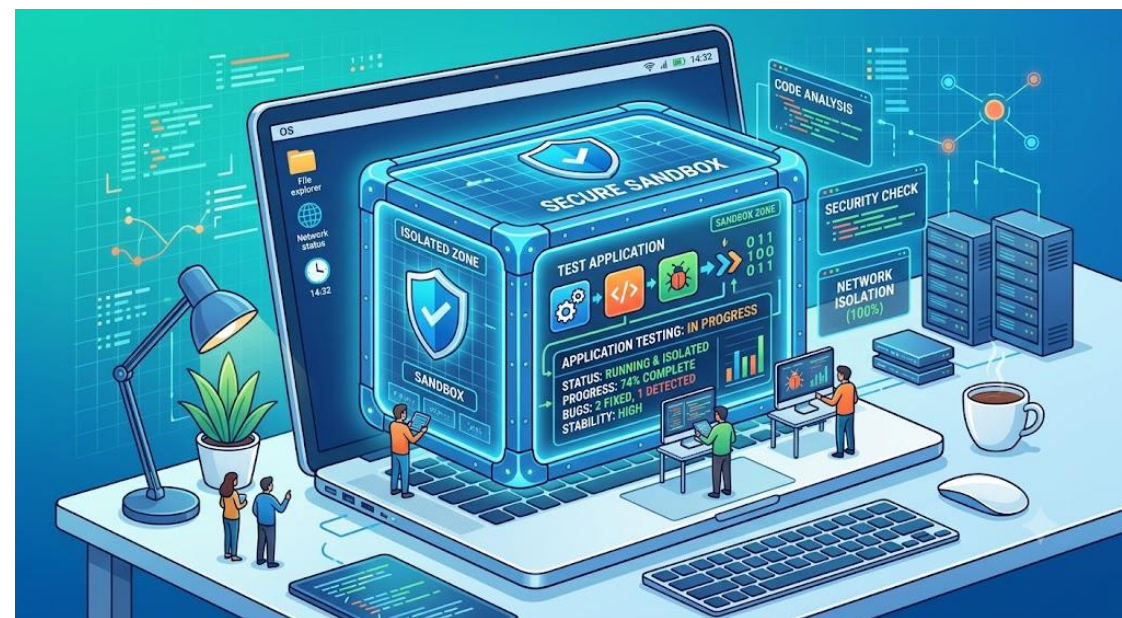
🌐 公開サイト（本番環境）

ユーザーが実際にアクセスする場所。
止められない、失敗が許されない本番ステージ。

🖥️ 検証環境（ローカル環境）

自分のPC内に作る安全な実験場。
何度壊しても誰にも迷惑がかからない安全地帯。

→ 安全な練習場所で動作確認をしてから本番へ適用します。



ローカル検証環境・3つの選択肢

本番サイトの状況や、ご自身のスキルレベルに合わせて最適なツールを選びましょう。



本日使用

1. Studio

最も手軽で超軽量
数秒で起動し初心者でも使いやすいツール



定番・万能

2. Local

多機能で環境切り替えの内容が豊富
通常の制作業務でも使い易いツール



エンジニア向け

3. wp-env

WordPressコアチーム公式
Dockerを用いたコマンド操作ツール

① Studio（スタジオ）

特徴とメリット

- 超軽量&高速起動：数秒でWordPressが立ち上がる
- シンプルなGUI：画面操作が極めて直感的で迷わない
- 低トラブル：PC環境によるエラーが非常に発生しにくい

おすすめ用途・検索ワード

サクッと行う単体検証や、ローカル環境での日常的なWordPress編集・テスト作業に最適です。

 推奨検索キーワード：「wp studio」または「WordPress Studio」

② Local（ローカル）

✓ 特徴とメリット

- Webサーバー切替：Apache / Nginx をワンクリック切り替え
- メール受信テスト：Mailpit機能が標準内蔵
- 複数サイト管理：多数の検証環境を個別に一元管理可能

👍 おすすめ用途・検索ワード

フリーランスやWeb制作会社での、日常的なクライアントサイト保守運用・複数サイト運用に最適。

🔍 推奨検索キーワード：「localwp」または「Local by Flywheel」

</> ③ wp-env (ダブルピー・エンプ)

✔ 特徴とメリット

公式開発ツール：WordPress公式チームが開発するコンテナ型ツール。

環境のコード化：.wp-env.json 設定ファイルを共有するだけで、チーム全員が全く同一のサーバー環境を再現できます。

⚠ 注意点とお勧め

注意点：Dockerのインストールが必要であり、ターミナル（黒い画面）の操作知識が必須です。

おすすめ：本格的なテーマ・プラグイン開発、複数人でのチーム開発プロジェクト。

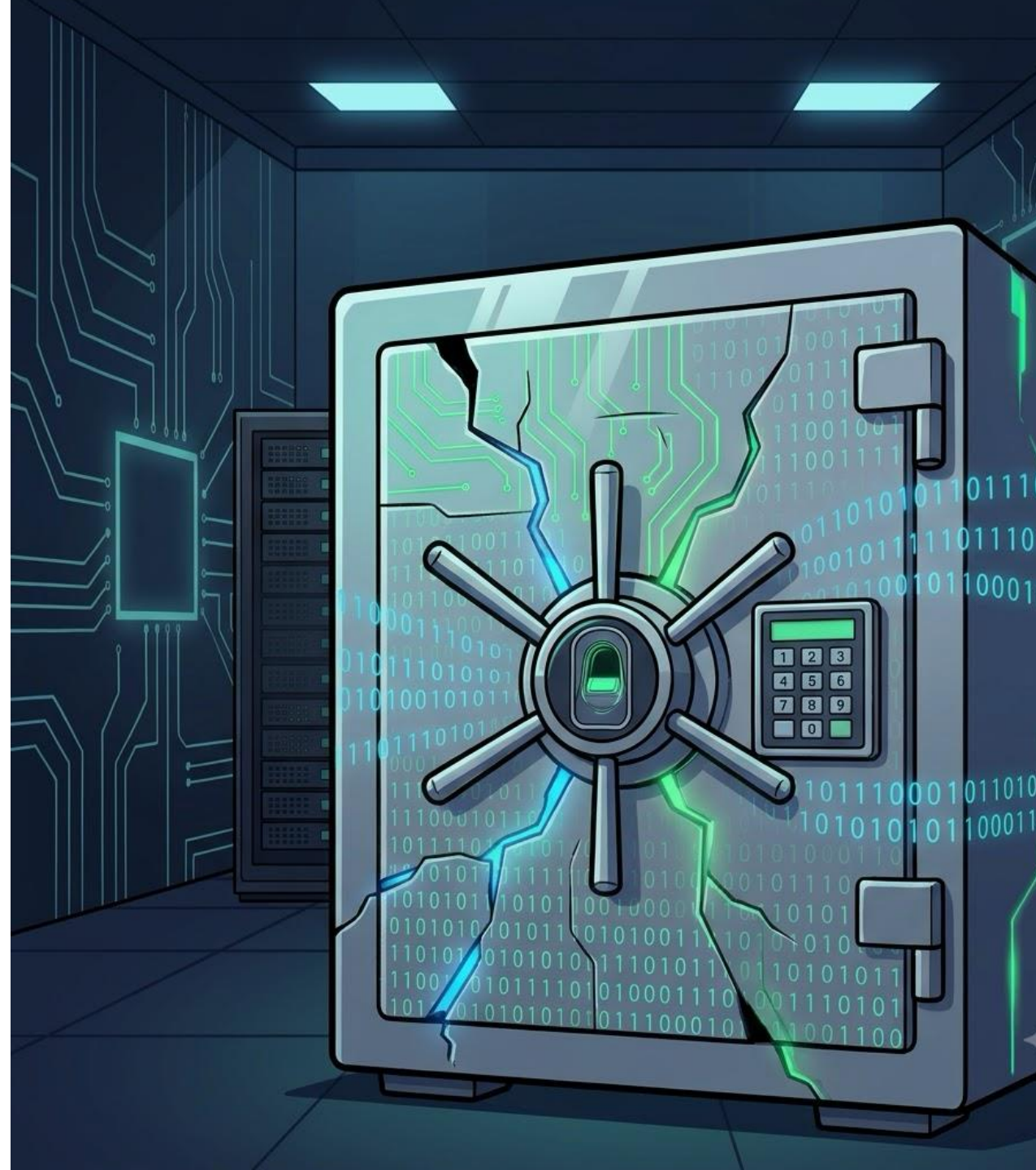
📌 必須作業：バックアップ

検証環境でもバックアップが必要な理由

ローカル環境であっても、作業に失敗して一から環境を作り直すのは大きな時間ロスになります。

要所ごとに「復元ポイント」を作っておくことで、失敗を恐れず大胆にアップデートのテストが実施できます。

何度でも試せる環境で納得できるまで、試して本番での作業に挑むことが重要です



バックアップ手段の選び方



プラグインはあくまで一例です

本日は参考例として「All-in-One WP Migration」をインストールしています。



実務での考慮ポイント

実際の現場ではサイトのデータ規模、ライセンス規約、商用利用制限などを考慮して選定します。



最も重要な基準

作業自身自身が**使い慣れている信頼性の高いプラグイン・手段**を採用することが最も安全です。

（例：BackWPup、UpdraftPlus、ホスティングサーバーの標準バックアップ機能など）

🔄 求めるバックアップ機能

どのツールを使う場合も、以下の基準を満たしていると安心・安全です。



ワンクリック復元

画面上の簡単な操作だけで、
データベースとファイルを丸ごと一括エクスポートできること



確実な巻き戻し

問題発生時、数ステップで「復元ポイント」
へ確実に巻き戻せること



チームでの共有性

サーバーの深い専門知識がなくても、
簡単に復元手順を実行できること。

■ 実践「3つの復元ポイント」

アップデート作業の各節目で、バックアップ（復元ファイル）を作成・保存します。

A 復元ポイントA（初期状態）

何も触っていない、
本番から預かったままの完全な初期状態

B 復元ポイントB（環境変更後）

プラグイン、テーマファイルなどをアップデートした時
PHPなどのサーバー環境を更新した時
など、何らかの変更後

C 復元ポイントC（完全体）

すべてをアップデート、更新し、
メール送信テスト等も完了した完成状態

🖥️ 【実演】 検証デモ環境の構成

📋 検証デモ環境の初期構成

- ~~WordPress~~ : ~~v5.9 (古い本体)~~
- ~~PHP~~ : ~~v7.4 (旧バージョン)~~
- ~~テーマ~~ : ~~Snow Monkey (例 : v16.0.0)~~
- ~~プラグイン~~ : ~~Snow Monkey Forms (例 : v4.0.0)~~
- ~~検証用~~ : ~~WP Mail Logging (送信確認用)~~

🧪 実験内容

まず「復元ポイントA」を作成した上で、このサイトのエンジン（PHP）と本体（WordPress）の更新作業を段階的に進めます。

どのような不具合（エラー）が発生し、どう修復するかをステップ・バイ・ステップで確認します。

⚠ 事象① : PHP 8.1の影響

~~現象 : サイレント不具合 (見た目は正常)~~

StudioでPHPを7.4→8.1へ変更。見た目のレイアウト崩れは発生しないが...
~~お問い合わせの「送信ボタン」を押すとアイコンが回ったままフリーズ!~~

解説 : PHP 8の記述ルール厳格化により通信(Ajax)が停止
——— 目視だけでは発見が遅れるタイプのエラー



※ 事象②：最新WordPressの影響

× 発生した現象

操作：WordPress本体とテーマを最新版へ一括アップデート
(※プラグインは古いまま放置)

不具合発生：お問合せページに「このブロックには無効なコンテンツが含まれています」とグレーのエラーが表示されフォームが消滅！

🔍 エラーの原因解説

最新のWordPressコアが要求するデータ構造と、放置された古いプラグインの構造が一致しなくなったため発生します。

ブロックエディター採用以降、よく発生するデータ不一致エラーの代表例です。

🔧 復旧作業と解決手順

📍万が一作業が難航しても、先ほど保存した「復元ポイントB」に戻せばいつでも安全にやり直せます！

- 1 プラグインの最新化：~~置いてけぼりだった~~「Snow Monkey Forms」を最新版に更新
- 2 ブロックの修復：編集画面を開き、WordPress標準機能の「ブロックを修復」をクリック
- ✅ 結果：フォームが正常に復旧し、送信テストも成功！→「**復元ポイントC**」を保存

【まとめ】アップデート手順

安全にバージョンアップを行うための、標準的なステップです。

プラグイン → テーマ

まずは周辺のプラグインおよびテーマを最新化する

WordPressコア

受け入れ態勢が整ったらWordPress本体を最新化

サーバー環境（PHP）

最後に土台となるサーバーのPHPバージョンを最新化

※各ステップの節目で必ずフロント画面表示とお問い合わせなどの動作テストを実施します。

? この順番で行う理由

周辺から中心に向かって固める

本体（コア）を先に上げてしまうと、古いプラグインやテーマが新しいコアについていけず、サイト全体が致命的にクラッシュします。

まず周辺を最新にして「**新コアを受け入れる状態**」を作ることが肝心です。

土台（PHP）を最後に更新

全てのWordPressシステム側を最新化した上で、最後に土台であるPHPを更新します。

これにより、システム全体の互換性を最も安全かつリスクなく担保することができます。

まとめ

- ✓ アップデートは「周辺（プラグイン・テーマ）から中心（コア）」の順番で
- ✓ 「手元の検証環境」と「使い慣れたバックアップ手段」で、いつでも安心して更新できる状態を目指す！

ご静聴ありがとうございました！