Engineer

バリデーション

入力を信頼する前に規則に合うか確かめること

どのようなものか

バリデーションは、プログラムが入力データを、それが従うべき規則と比較し、満たさなければ拒否する段階です。保存、実行、転送より前に行います。層によって呼び名が異なります。フォームの欄やAPIリクエストなら入力検証、規則をJSON SchemaやOpenAPIの記述で定めるならスキーマ検証、WebページをHTMLの文法と照合するならマークアップ検証です。検査に通った入力を、有効と呼びます。

OWASPの指針は、二つの問いに分けています。構文の検査は、期待する型と形式かを問います。日付か、整数か、本文をJSONとして解析できるか、といったことです。意味の検査は、「その操作にとって、値に意味があるか」を問います。ホテルの予約なら、有効な日付が二つあるだけでなく、終了日が開始日より後である必要があります。文字列"500"を整数へ変換しても、一つ目の問いに答えただけです。500泊の滞在を認めるかは分かりません。

マークアップ検証は、Web上で最も早く公開された、この考え方の実例です。W3C Markup Validation Serviceは、文書が使うと宣言した文法と照合することだと説明し、自身のバリデーターをCのlintになぞらえています。その解説文書は、何を証明できるかについて慎重です。有効なページは、必ずしも良いページではなく、ある規則群に適合するとツールが判断したページにすぎません。同じ注意は、ほかの検証にも当てはまります。バリデーションで分かるのは、データが書いた規則に合うことです。規則自体が正しいかは分かりません。

近い二つの考え方と、よく混同されます。サニタイズは、削除や書き換えによって入力を受け入れられる形へ変えます。バリデーションは、入力をそのまま受け入れるか拒否します。符号化とパラメーター化は、HTMLページやSQLクエリーなど、送り先でデータを安全に扱える形にします。バリデーションに通った後も必要です。

概念を判断に生かす

「バリデーションとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 取り込み時に不適切なデータを検出する規則と検査を設け、各数値の責任者と出所を明らかにします。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 14部門を支える共通インフラ計画 — 14 共通インフラ計画の対象となった研究部門

14部門を支える共通インフラ計画 お問い合わせ →

なぜ重要なのか

バリデーションで決めるのは、システムのどこに信頼境界を引き、そこで何を検査するかです。ブラウザーのリクエスト、キューのメッセージ、取引先のファイル、同僚による設定変更には、それぞれ前提があります。入口で前提を確かめれば、残りのコードはそれに頼れます。省略すれば、後段の各関数が自身を守る必要がありますが、実際には多くがそうしません。

二つ目の判断は、規則の書き方です。エンドポイントごとの手書きの検査は、最初は安上がりでも、システムが育つと食い違います。それぞれが独立した小さなコードだからです。型やスキーマとして一度だけ宣言した規則なら、ライブラリで強制し、同じ情報から文書も公開できます。説明と動作が食い違うリスクを減らせます。その代わり、フレームワークを学び、既定の動作を理解する必要があります。スキーマが強制するのは、明記したことだけであり、書かなかったことは強制しないからです。

最後に選ぶのは厳しさです。許可リストは、受け入れるものを記述し、それ以外をすべて拒否します。OWASPは、悪意のある文字列をすべて見分けようとする方法より、こちらを勧めています。スキーマが知らないフィールドを拒否すれば、入力ミスや古いクライアントを早く見つけられます。最初の問題でリクエスト全体を明確なエラーとともに拒否する方が、一部しか検査していないデータで続けるより安全です。

実際の活用例

あるAPIは、エンドポイントごとにリクエスト検証を手書きしていました。記録された実装段階で、Huma v2へ移し、649パス、760操作の検査を型で表現しました。各エンドポイントに入力と出力の構造体を持たせ、Humaがリクエスト検証を生成し、OpenAPIの記述も同じ型から作りました。WebSocketの二つのエンドポイントは、そのリクエスト・レスポンスモデルに合わず、素のFiberに残し

ました。CIは、サーバーが登録する経路と記述が示す経路を比較しました。記録された経路の一致率は100%で、入力型で宣言した検査が、対応する手書きの検証処理に置き換わりました。

設定にも同じ扱いが必要です。会社のホストのSSH堅牢化では、収束処理のたびに三段階の検証を行います。候補のファイルを設置前に構文検査し、設置後に組み上がった設定を検査し、最後にデーモンが解決した実効設定を読み戻し、24の設定項目を照合します。三段階目があるのは、2026年9月のアクセス報告で、自動化が毎回書いていた二つの設定を、デーモンが上書きしていたと分かったからです。OS自身の既定値を先に読み、その二つは最初の出現が優先されていました。リンク先の言葉どおり、grepで設定項目を見つけても、書かれたことしか証明できません。

マークアップ検証も、記録された2026年8月の実装段階に含まれていました。22リンターを使うコミットの検査に、HTML検証を組み込み、警告という別扱いを設けず、どの診断もエラーとし、行内の抑制を禁じていました。そのため、エラーを許容扱いにして、不正なページをコミットすることはできませんでした。

問題が起きるところ

サーバーが、ブラウザーで検査済みだと信じる。 ブラウザーで不正なメールアドレスを拒否するフォームは、検証しているように見えます。しかし、フォームを使わずにリクエストを送れます。OWASPも、クライアントだけの検査を典型的な落とし穴に挙げています。検証はサーバーで行い、ブラウザーの検査は入力する人への素早い案内として扱います。

スキーマが、明記しないものをすべて受け入れる。 JSON Schemaでプロパティーを列挙しても、必須にはならず、ほかのフィールドも禁じません。また、既定の2020-12語彙では、 `format` は注釈として収集され、形式の検査は既定で無効です。次のスキーマは、厳しく見えます。

```
{
    "type": "object",
    "properties": {
        "email": { "type": "string", "format": "email" }
    }
}
```

しかし、`{}`、`{"email": "nope"}`、`{"emial": "a@example.com"}`を受け入れます。`required`と`"additionalProperties": false`を追加し、バリデーターで形式の検証を有効にするか、コードで形式を検査します。一方、Humaは、未知の本文フィールドを既定で拒否し、`readOnly`と`writeOnly`を文書には示しても強制せず、別途指定しない限り未知のクエリーパラメーターを無視します。

検査したデータと、プログラムが使うデータが違う。 バリデーターが生のリクエストを読み、ハンドラーがデコードしたコピーを読む場合や、テストがディスク上のファイルを調べ、デーモンが別の設定を使う場合、システムが使わないものを検査しています。OWASPは、「実際に使う表現を検証する」と助言しています。上のSSHの仕事も、設定について同じ教訓を示しています。実際に適用された値は、デーモンから読み戻して初めて分かりました。

拒否リストが、次の変種に出会う。 既知の悪い文字を拒否すると、姓のアポストロフィーなど正当な入力も拒みながら、クエリーはなお危険なままです。フィールドごとに受け入れるものを定義し、残りを拒否します。

有効な入力を、安全な入力と見なす。 規則をすべて満たした文字列も、SQL文やページにそのまま貼り付ければ壊せます。バリデーションは、届くものを限定します。パラメーター化したクエリーと出力時の符号化は、行き着く先を守ります。両方が必要です。

参考資料

- OWASP Input Validation Cheat Sheet — 構文と意味、許可リスト、解析上の上限、よくある落とし穴を扱います。
- W3C Markup Validatorの文書 — マークアップ検証についてのバリデーター自身の説明と、有効性が証明すること・しないことのヘルプです。
- JSON Schema Validation、draft 2020-12 — `format`の既定の動作を含む、検証用キーワードの仕様です。
- Huma：Request Validation — Go構造体のタグから生成する検証と、フィールドについて厳しくする設定・緩くする設定です。
- OpenAPI Specification — 多くのAPIバリデーターがスキーマを検査に使う、記述の形式です。

用語集

実際の制約に合わせて選ぶ

代表的なデータセット、そこから答えを得たい問い、現在のボトルネックをお知らせください。品質、処理量、復旧の測り方を合意できます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「バリデーションとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/validation/