Eingaben validieren und casten

Wie PHP-Code in Exponential aus nicht vertrauenswürdigen Eingaben sichere Werte macht - Lesen über eZHTTPTool und $Params, Casts mit (int) und (float) samt Bereichen, Allow-Listen mit striktem in_array, Regeln für…

Source file: doc/ger-de/hilfe/php-entwickler/eingaben-validieren-und-casten.md Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/php-developers/validating-input-and-casting.md

Diese Seite richtet sich an PHP-Entwickler, die Werte aus Anfragen, Einstellungen, Importen oder der Datenbank übernehmen. Der eigene Weg des Kernels ist in jeder View sichtbar: über die Hilfsmittel des Kernels lesen, den Wert in den Typ umwandeln, den er haben muss, ihn gegen das Erlaubte prüfen und ihn erst dann verwenden. Da der Code-Stil keine Typdeklarationen kennt, sind es diese Casts, die einem Wert seinen Typ geben (Warum der Code keine Typdeklarationen hat).

Inhalt

1. Eingaben lesen
2. Zahlen
3. Auswahlwerte
4. Text
5. E-Mail-Adressen und URLs
6. Arrays aus der Anfrage
7. Fehler an den Benutzer melden
8. Eine Checkliste

Eingaben lesen

PHP

```php
$http = eZHTTPTool::instance();
$title = $http->hasPostVariable( 'Title' ) ? (string)$http->postVariable( 'Title' ) : '';
$page  = $http->hasGetVariable( 'page' ) ? (int)$http->getVariable( 'page' ) : 1;
$id    = isset( $Params['ItemID'] ) ? (int)$Params['ItemID'] : 0;
```

Quelle	Lesen mit
POST	`$http->hasPostVariable()`, `$http->postVariable( $name, $fallback )`
GET	`$http->hasGetVariable()`, `$http->getVariable( $name, $fallback )`
beides	`$http->variable( $name, $fallback )`
View-Parameter	`$Params['Name']` (`params` und `unordered_params` aus `module.php`)
Formular-Schaltflächen	`$Module->isCurrentAction()`, `$Module->actionParameter()`
Uploads	`eZHTTPFile::canFetch()`, `eZHTTPFile::fetch()`

Jede davon gibt Strings, Arrays oder `null` zurück. Unter Velocity bereitet der Kernel die Anfrage pro Worker-Anfrage vor; wer `$_POST` und `$_GET` direkt liest, umgeht das.

Zahlen

PHP

```php
$id     = (int)$Params['ItemID'];                                  // ids
$limit  = max( 1, min( 100, (int)$http->getVariable( 'limit', 20 ) ) );   // bounded
$offset = max( 0, (int)$http->getVariable( 'offset', 0 ) );
$amount = (float)str_replace( ',', '.', (string)$http->postVariable( 'Amount', '0' ) );
```

- `(int)` macht aus allem eine Ganzzahl (`'12abc'` wird 12, `'abc'` wird 0): Prüfen Sie den Bereich nach dem Cast.
- Eine ID von `0` bedeutet „nicht angegeben“: Behandeln Sie sie als nicht gefunden.
- Begrenzen Sie jedes Limit: Ein unbegrenztes `limit` ist eine Dienstverweigerung (Denial of Service).

Auswahlwerte

PHP

```php
$sort = (string)$http->getVariable( 'sort', 'created' );
if ( !in_array( $sort, array( 'created', 'title', 'amount' ), true ) )
    $sort = 'created';
```

Übergeben Sie immer `true` als drittes Argument von `in_array()`: Ohne es ist der Vergleich lose, sodass `'1e1'` zu `'10'` passt, `null` zu `''` und `true` zu jedem nicht leeren String. Verwenden Sie Allow-Listen für alles, was zu einem Spaltennamen, einer Sortierrichtung, einem Template-Namen, einem Klassennamen oder einem Dateinamen wird.

Text

PHP

```php
$title = trim( (string)$http->postVariable( 'Title', '' ) );
if ( $title === '' )
    $errors['title'] = ezpI18n::tr( 'extension/acmereport', 'Enter a title.' );
else if ( mb_strlen( $title, 'UTF-8' ) > 200 )
    $errors['title'] = ezpI18n::tr( 'extension/acmereport', 'The title can have at most 200 chara
cters.' );
```

- Casten Sie zuerst zu String (`(string)`), damit `null` und Arrays nie String-Funktionen erreichen (eine Deprecation in PHP 8.1 und ein `TypeError` bei Arrays).
- Zählen Sie Zeichen mit `mb_strlen()`, nicht Bytes.
- Bewahren Sie Text so, wie der Benutzer ihn geschrieben hat; maskieren Sie bei der Ausgabe, nicht beim Speichern.
- Für Bezeichner (einen Slug, einen Code) prüfen Sie gegen ein Muster: `preg_match( '/^[a-z0-9_-]{1,64}$/', $code )`.

E-Mail-Adressen und URLs

PHP

```php
$email = trim( (string)$http->postVariable( 'Email', '' ) );
$validEmail = filter_var( $email, FILTER_VALIDATE_EMAIL ) !== false;

$url = trim( (string)$http->postVariable( 'Website', '' ) );
$scheme = strtolower( (string)parse_url( $url, PHP_URL_SCHEME ) );
$validUrl = filter_var( $url, FILTER_VALIDATE_URL ) !== false && in_array( $scheme, array( 'http'
, 'https' ), true );
```

Erlauben Sie für URLs, die Sie verlinken oder abrufen, nur `http` und `https`. Weiterleitungsziele haben ihre eigenen Regeln (Sicherheit in PHP-Code).

Arrays aus der Anfrage

Ein Feld kann als Array ankommen (`Title[]=x`), selbst wenn Ihr Formular einen String sendet. Prüfen Sie den Typ:

PHP

```php
$ids = $http->postVariable( 'SelectedIDs', array() );
$ids = is_array( $ids ) ? array_values( array_filter( array_map( 'intval', $ids ) ) ) : array();
```

Fehler an den Benutzer melden

Sammeln Sie Fehler pro Feld, rendern Sie das Formular mit den Werten neu, die der Benutzer eingegeben hat, und zeigen Sie die Fehler neben den Feldern und in einer Zusammenfassung oben an. Datentypen melden stattdessen über den Validator des Kernels: `$objectAttribute->setValidationError( ... )` und `return eZInputValidator::STATE_INVALID;` (Einen Datentyp schreiben).

Eine Checkliste

- ☐ Jeder Wert über `eZHTTPTool`, `$Params` oder die Aktionen des Moduls gelesen.
- ☐ Jede Zahl gecastet und begrenzt.
- ☐ Jeder Auswahlwert mit striktem Vergleich gegen eine Allow-Liste geprüft.
- ☐ Jeder String gecastet, getrimmt und in Zeichen auf seine Länge geprüft.
- ☐ Jedes Array vor der Verwendung mit `is_array()` geprüft.
- ☐ Fehler dem Benutzer angezeigt, ohne rohe Eingaben wiederzugeben.

Verwandte Seiten

- Eingabevalidierung und Ausgabemaskierung
- eZHTTPTool-Referenz
- Härtung der Eingaben von Datentypen
- Freie Dokumentation: diese Seite darf unter der GNU FDL 1.3 frei kopiert, übersetzt und verbessert werden.

Werden Sie Exponential-Partner

Partner teilen ihre Informationen, stellen Mittel bereit, geben Beiträge zurück und entwickeln die Produkte mit uns, und so wächst Exponential. Eine Partner-Website und ein Dashboard sind für Mitte 2027 geplant: Werden Sie noch heute Partner, damit wir schneller wachsen und neue Funktionen früher liefern.

Partner: hier beginnen | Wie Sie sich bewerben

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.