Die Debug-Leiste für PHP-Entwickler

Die Exp Debug-Leiste als PHP-Entwickler nutzen - die Reiter (Meldungen, Einstellungen, Cache, Zeiten, SQL, Templates, Eingebundene Dateien, Speicher, Rendering, Benutzer, Server), die Schwellenwerte der Zusammenfassung…

Source file: `doc/ger-de/hilfe/php-entwickler/die-debug-leiste-fuer-php-entwickler.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/php-developers/the-debug-bar-for-php-developers.md

Diese Seite richtet sich an PHP-Entwickler, die den Debug-Bericht am unteren Rand einer Seite verwenden. Seit 6.0.15 ist das die **Exp Debug-Leiste** (`lib/ezutils/classes/expdebugbarreport.php`): eine angeheftete Leiste mit einer Live-Zusammenfassung der Anfrage und einem Panel mit Reitern. Die vollständige Referenz ist Die Exp Debug-Leiste; diese Seite behandelt, was ein PHP-Entwickler täglich verwendet und wie Sie ein eigenes Panel hinzufügen.

Inhalt

1. Einschalten
2. Die Zusammenfassung
3. Die Reiter
4. Einstellungen aus der Leiste ändern
5. Ein eigenes Panel
6. Unter Velocity

Einschalten

`site.ini [DebugSettings] DebugOutput=enabled`, beschränkt auf Ihre Adresse mit `DebugByIP=enabled` und `DebugIPList[]`, auf einer Entwicklungs- oder Testinstallation (Debugging mit eZDebug). Um Einstellungen zu ändern und Caches aus der Leiste zu leeren, brauchen Sie die Richtlinie `setup/setup` (Cache-Steuerung: `setup/managecache`).

Die Zusammenfassung

Die Leiste zeigt, was `eZDebug` für die Anfrage gemessen hat: Seitenzeit, Anzahl und Zeit der SQL-Anweisungen, Spitzenspeicher und das Limit, verwendete Templates, Meldungen pro Stufe, die Engine (Velocity oder die SAPI), den Siteaccess und, wenn er eingeschaltet ist, den SQL-Abfrage-Cache. Jeder Wert bekommt eine Stufe, `ok`, `warn` oder `high`, gemessen an `Thresholds[]` in `settings/debugbar.ini` (`time_ms`, `sql_count`, `sql_ms`, `memory_mb`, `templates`, als `<warn>;<high>`). Ein roter Wert auf einer Seite, die Sie geändert haben, ist der erste Hinweis auf ein Performance-Problem.

Die Reiter

Reiter	Was ein PHP-Entwickler dort liest
Meldungen	die Hinweise, Warnungen und Fehler, die Ihr Code und der Kernel geschrieben haben, nach Stufe filterbar
Einstellungen	jede registrierte Debug-Einstellung, ihr geltender Wert und die Datei, aus der er stammt; je Geltungsbereich änderbar
Cache	diese Seite, Inhalte, Templates, INI oder alles leeren
Zeiten	Zeitpunkte und Akkumulatoren, Ihre eingeschlossen (PHP-Code profilieren)
SQL	Anzahl der Abfragen, Zeit, Anteil an der Seite und die Anweisungen (mit `SQLOutput=enabled`), die langsamsten zuerst
Templates	die verwendeten Templates (`ShowUsedTemplates`) und der Bericht des Packers
Eingebundene Dateien	die eingebundenen PHP-Dateien (`DisplayIncludedFiles`)
Speicher	aktueller Speicher, Spitze, Limit und die Schritte, die am meisten verbraucht haben
Rendering	der Controller, der View-Cache für Inhalte und sein Schlüssel, die kompilierten Templates (für `setup/setup`)
Benutzer	das Cache-Profil des Benutzers (ob zwei Benutzer sich die gecachten Ansichten teilen), Präferenzen, Rollenzuweisungen und auf Anfrage die Richtlinien dieser Rollen, nach Rolle und per Suche (für jeden, der die Debug-Ausgabe erhält, und `setup/setup`)
Server	wo die Anfrage eingegangen ist und welcher Prozess sie gerendert hat (Velocity, PHP-FPM, mod_php, die Kommandozeile), OPcache-Status, der Antwort-Cache von Velocity

Das Filterfeld über den Reitern filtert die Zeilen aller Reiter zugleich.

Einstellungen aus der Leiste ändern

Der Reiter Einstellungen schreibt über die Engine `exp:ini`, im Geltungsbereich Ihrer Wahl (globale Überschreibung, ein Siteaccess oder eine Erweiterung). Jeder Schreibvorgang ist eine JSON-Zeile in `var/<site>/log/debugbar.log` und lässt sich aus der Leiste rückgängig machen, solange die Einstellung seither nicht erneut geändert wurde. Debug-Bedingungen aus `debug.ini` `[GeneralCondition]` und Debug-Schalter aktiver Erweiterungen (eine Variable, deren Name „debug“ enthält, mit einem booleschen Wert) werden automatisch gefunden und als Schalter angezeigt.

Ein eigenes Panel

Eine Erweiterung kann einen Reiter mit Daten über ihren Teil der Anfrage hinzufügen. Registrieren Sie ihn in `extension/acmereport/settings/debugbar.ini.append.php`:

INI

```
<?php /* #?ini charset="utf-8"?
[DebugBarSettings]
Panels[]=acmereport

[Panel_acmereport]
Class=acmeReportDebugBarPanel
*/ ?>
```

und implementieren Sie `expDebugBarPanelInterface` (`kernel/classes/debugbar/expdebugbarpanelinterface.php`):

PHP

```php
class acmeReportDebugBarPanel implements expDebugBarPanelInterface
{
    public function visible()
    {
        // the bar also shows to anonymous visitors when DebugByIP matches
        $access = eZUser::currentUser()->hasAccessTo( 'acmereport', 'read' );
        return $access['accessWord'] !== 'no';
    }

    public function title()
    {
        return ezpI18n::tr( 'extension/acmereport', 'Reports' );
    }

    public function chip()
    {
        return array( 'text' => 'Reports: ' . acmeReportStats::count(), 'title' => 'Items read in
 this request', 'level' => 'ok' );
    }

    public function html()
    {
        return '<p>' . htmlspecialchars( acmeReportStats::summary(), ENT_QUOTES | ENT_SUBSTITUTE
| ENT_HTML401, 'UTF-8' ) . '</p>';
    }

    public function data()
    {
        return array( 'items' => acmeReportStats::count() );
    }
}
```

Regeln aus der Referenz der Leiste:

- `visible()` wird zuerst gefragt; ein Panel, das nein antwortet, zeichnet nichts.
- `html()` wird so ausgegeben, wie es ist: Maskieren Sie alles, was Sie hineinschreiben. Text und Titel des Chips maskiert die Leiste.
- Die Klasse wird für jeden Bericht erzeugt, auch für anonyme Besucher: Halten Sie den Konstruktor trivial und die Methoden günstig, ohne eigenen Datenzugriff; sammeln Sie Werte, während die Anfrage läuft, und lesen Sie sie hier nur.
- Ein Panel, das eine Exception wirft, wird mit einer Warnung ausgelassen; der Rest der Leiste wird gezeichnet.
- `data()` geht in das JSON der Seite unter `panels.<id>`.

Unter Velocity

Die Leiste nennt die Engine im Reiter Server. Wenn ein Wert nach einer Codeänderung falsch aussieht, prüfen Sie, ob die Worker neu gestartet wurden (`exp:velocity deploy`): Worker behalten die Klassen, die sie geladen haben. Ein Engine-Archiv ohne `expdebugbarreport.php` gibt statt der Leiste den klassischen Bericht aus (Persistente Velocity-Worker für PHP-Entwickler).

Verwandte Seiten

- Exp Debug-Leiste (Funktion)
- Erweiterungspunkt Panel der Debug-Leiste
- Erweiterungen debuggen
- 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.