Einen Konsolenbefehl schreiben

Eine Anleitung für ein Kommandozeilenwerkzeug in einer Erweiterung - die Befehlsklasse, die Exponential\Runnable\Command erweitert, die Einstiegsdatei, Optionen und Hilfe, Siteaccess und Datenbank, Ausgabe und…

Source file: doc/ger-de/hilfe/php-entwickler/einen-konsolenbefehl-schreiben.md Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/help/php-developers/write-a-console-command.md

Diese Anleitung richtet sich an PHP-Entwickler, die einer Erweiterung ein Kommandozeilenwerkzeug hinzufügen: einen Import, einen Bericht, ein Reparaturskript. Der Befehl ist eine Klasse, die `Exponential\Runnable\Command` erweitert; eine kurze Einstiegsdatei ruft sie auf; die Konsole `bin/php/console` findet sie ohne jede Registrierung.

Inhalt

1. Das Ergebnis
2. Schritt 1: die Befehlsklasse
3. Schritt 2: die Einstiegsdatei
4. Schritt 3: Autoloads und ein erster Lauf
5. Optionen und Argumente
6. Die Hilfsmethoden
7. Ausgabe, Logging und Exit-Codes
8. Ausführung als root, Siteaccesses und Benutzer
9. Lang laufende Befehle

Das Ergebnis

Shell

```
php bin/php/console ext:acmereport:report --month=2026-09
php bin/php/console list ext:acmereport            # every command of the extension
php bin/php/console ext:acmereport:report --help
```

Schritt 1: die Befehlsklasse

`extension/acmereport/classes/runnable/commands/report.php`:

PHP

```
<?php
namespace Exponential\Command\Extension\Acmereport;

class Report extends \Exponential\Runnable\Command
{
    public function run()
    {
        $this->script( array( 'description' => "Writes the monthly report of acmereport",
                              'use-session' => false, 'use-modules' => true, 'use-extensions' =>
```

```
true ) );
        $options = $this->startup( '[month:]', '', array( 'month' => 'The month, YYYY-MM' ) );

        $month = (string)$options['month'];
        if ( !preg_match( '/^\d{4}-\d{2}$/', $month ) )
        {
            $this->error( 'Give the month as --month=YYYY-MM' );
            $this->shutdown( 1 );
        }

        $totals = \acmeReportService::monthlyTotals( $month );   // the work: a class you can cal
l anywhere
        foreach ( $totals as $status => $total )
            $this->output( sprintf( '%-12s %8d', $status, $total ) );

        $this->shutdown( 0 );
    }
}
```

Schritt 2: die Einstiegsdatei

`extension/acmereport/bin/php/report.php`:

PHP

```
#!/usr/bin/env php
<?php
/**
 * @description Writes the monthly report of acmereport
 */
require_once 'autoload.php';
\Exponential\Command\Extension\Acmereport\Report::main( __FILE__ );
```

Das Tag `@description` ist das, was `php bin/php/console list` anzeigt; ein Tag `@long-description` fügt der Hilfe Text hinzu. Machen Sie die Datei ausführbar (`chmod +x`).

Schritt 3: Autoloads und ein erster Lauf

Shell

```
php -l extension/acmereport/classes/runnable/commands/report.php
php bin/php/ezpgenerateautoloads.php -e
php bin/php/console ext:acmereport:report --help
php bin/php/console ext:acmereport:report --month=2026-09
```

Führen Sie jeden Befehl aus dem Wurzelverzeichnis der Installation aus (dem Verzeichnis mit `index.php`); `require_once 'autoload.php'` verlässt sich darauf.

Optionen und Argumente

Der Options-String hat das Format von `eZCLI::parseOptionString()`, das `eZScript::getOptions()` verwendet:

Form	Bedeutung
`[o]`	eine kurze Option, `-o`
`[opt]`	eine lange Option, `--opt`

Form	Bedeutung
`[o|opt]`	dieselbe Option in kurzer oder langer Form
`[opt:]`	eine Option mit Wert, `--opt=value`
`[opt;]`	ein optionaler Wert, `--opt` oder `--opt=value`
`[opt*]`, `[opt+]`, `[opt?]`	beliebig oft, mindestens einmal, höchstens einmal angegeben

Lesen Sie die Werte aus dem zurückgegebenen Array über den langen Namen (`$options['month']`), und prüfen Sie eine Option, die fehlen kann, mit `empty()`, bevor Sie sie verwenden.

Das zweite Argument beschreibt Positionsargumente; das dritte liefert den Hilfetext jeder Option. Die Standardoptionen gibt es gratis: `--help`, `-q`/`--quiet`, `-s`/`--siteaccess`, `-l`/`--login`, `-d`/`--debug`, `-v`/`--verbose`.

Die Hilfsmethoden

Hilfsmethode	Tut
`$this->script( $settings )`	erzeugt das Skript, `eZScript::instance( $settings )`
`$this->startup( $config, $argumentConfig, $optionHelp )`	`startup()` des Skripts, die Optionen, `initialize()` des Skripts (Siteaccess, Erweiterungen, Sitzung, Datenbank)
`$this->options( ... )`	nur das Parsen der Optionen, wenn Sie Schritte dazwischen brauchen
`$this->cli()`	das Terminal, `eZCLI::instance()`
`$this->output()`, `error()`, `warning()`	eine Zeile, ein Fehler, eine Warnung; `output()` schreibt mit `-q` nichts
`$this->isQuiet()`	`-q` wurde angegeben
`$this->log( $message, $logName = null )`	`eZLog::write()`, standardmäßig nach `var/log/<script name>.log`
`$this->shutdown( $exitCode, $exitText )`	`eZScript::shutdown()`, das beendet

Ausgabe, Logging und Exit-Codes

- Beenden Sie mit `0` bei Erfolg und einem anderen Code bei einem Fehlschlag: Deploy-Skripte, Cron und CI lesen ihn, und die Konsole gibt ihn unverändert weiter.
- Schreiben Sie Ergebnisse auf die Standardausgabe und Probleme über `error()`, sodass eine Pipe nur die Ergebnisse erhält.
- Protokollieren Sie, was Daten geändert hat (`$this->log()`), niemals Passwörter, Tokens oder personenbezogene Daten.
- Bieten Sie `--dry-run` für alles an, was Daten ändert.

Ausführung als root, Siteaccesses und Benutzer

- Skripte verweigern die Ausführung als `root`, sofern nicht `--allow-root-user` angegeben ist. Führen Sie sie nach Möglichkeit als Benutzer der Site aus, damit die Dateien, die sie in `var/` schreiben, der Site gehören.
- `-s <siteaccess>` wählt die Einstellungen; ohne die Option wird der Standard-Siteaccess verwendet.
- In einem Skript ist der aktuelle Benutzer anonym, bis das Skript einen anmeldet. Verwenden Sie einen eigenen Benutzer mit genau den Richtlinien, die der Befehl braucht, nicht den Administrator.

Lang laufende Befehle

- Holen Sie Inhalte seitenweise und leeren Sie die In-Memory-Caches zwischen den Seiten (`eZContentObject::clearCache()`), sonst wächst der Speicher mit jedem Objekt.
- Committen Sie in Stapeln mit kurzen Transaktionen.
- Geben Sie den Fortschritt aus, sofern nicht `-q`; Cron läuft mit `-q`.
- Befehle, die nach Zeitplan laufen sollen, sind besser ein Cronjob-Teil, der denselben Dienst aufruft (Einen Cronjob-Teil schreiben).

Verwandte Seiten

- Modul-Views als Klassen
- Exponential Console
- Konsolenbefehle für Entwickler
- 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.