Git-Arbeitsablauf

Wie Git für Exponential verwendet wird - main als einzige Entwicklungslinie, Branches und Pull-Requests, die zusammengefasst und neu aufgesetzt werden, solange sie offen sind, Merge-Commits, kein Umschreiben der…

Source file: `doc/ger-de/hilfe/softwareingenieure/git-arbeitsablauf.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/software-engineers/git-workflow.md

Diese Seite richtet sich an Ingenieure, die in Exponential committen, seine Geschichte prüfen oder Installationen aus Git-Klonen betreiben. Der Standard ist Quellcodeverwaltung und Releases; die praktischen Anleitungen sind Mit Git entwickeln und Mit Git aktualisieren.

Inhalt

1. Branches und Merges
2. Commit-Nachrichten
3. Hooks
4. Was nie in einen Commit gehört
5. Installationen aus Klonen
6. Bekannte Lücken

Branches und Merges

- `main` **ist die Entwicklungslinie**, für Composer mit dem Alias `6.0.x-dev`; die CI läuft darauf.
- **Gearbeitet wird auf Branches**, die per Pull-Request nach `main` gelangen.
- **Solange ein Pull-Request offen ist**, fasst sein Autor ihn zu logischen Commits zusammen (meist einem) und setzt ihn auf `main` neu auf, wie die Reviews es verlangen:

  Shell

  ```
  git fetch origin
  git rebase -i origin/main
  git push --force-with-lease
  ```

  Erzwingen Sie einen Push nur auf den Branch Ihres eigenen Pull-Requests.
- **Pull-Requests werden mit einem Merge-Commit zusammengeführt.** Nach dem Merge wird die Geschichte von `main` nie umgeschrieben; spätere Korrekturen kommen als neue Pull-Requests.

Commit-Nachrichten

Die Betreffzeile beginnt mit dem, was die Änderung tut:

Präfix	Für
Added:	eine neue Funktion, Datei, Klasse, Einstellung oder Fähigkeit

Präfix	Für
`Updated:`	eine Änderung an bestehendem Verhalten, Templates, Einstellungen oder Code
`Fixed:`	eine Fehlerkorrektur
`Removed:`	etwas Gelöschtes
`Renamed:`	eine umbenannte Datei, Klasse, Einstellung oder Erweiterung

Beispiel aus dem Hook: `Fixed: The redirect keeps the query string on the own host`. Die Betreffzeilen `Merge`, `Revert`, `fixup!`, `squash!` und `amend!` von Git selbst werden akzeptiert.

Der Text sagt warum und was, für einen Leser, der den Zusammenhang nicht kennt. Weitere Regeln, die kein Werkzeug prüft: Nennen Sie das Produkt Exponential (Bezeichner wie `eZContentObject` bleiben), keine Zuschreibung an Werkzeuge oder Assistenten, keine privaten oder unveröffentlichten Pfade, und vermischen Sie keine Änderungsarten in einem Commit, wenn sie sich trennen lassen.

Eine schlechte Betreffzeile eines Projektautors lässt die Prüfung scheitern; die eines Beitragenden erzeugt nur eine Warnung. Um Ihren Branch streng zu prüfen:

Shell

```
EXP_COMMIT_MSG_STRICT_ALL=1 sh .githooks/check-commit-range origin/main HEAD
```

Hooks

Shell

```
make hooks
```

setzt `core.hooksPath` auf `.githooks`: Der Pre-Commit-Hook prüft die Syntax (auch mit PHP 8.0, wenn vorhanden), das Dateimanifest und den Codestil der vorgemerkten Dateien; der Commit-Msg-Hook prüft die Betreffzeile; der Pre-Push-Hook führt PHPStan aus, wenn die Entwicklungswerkzeuge installiert sind. Überspringen Sie sie nur im Notfall (`--no-verify`); die CI führt dieselben Prüfungen aus (Statische Analyse und Codestil).

Was nie in einen Commit gehört

`.gitignore` schließt vieles davon bereits aus; prüfen Sie trotzdem:

- Zugangsdaten und Maschineneinstellungen: `settings/override/`, `config.env.php`;
- Laufzeitverzeichnisse: `var/cache/`, `var/log/`, `var/storage/`;
- `vendor/`, `composer.lock`, `.devtools/`;
- private Arbeitsnotizen.

Merken Sie Dateien namentlich vor statt mit `git add -A`, damit nichts Unversioniertes hineinrutscht.

Installationen aus Klonen

Composer kann den Kernel und die Erweiterungen als Git-Klone installieren. Dann sehen Sie, was sich geändert hat, finden, wann und warum sich eine Zeile geändert hat, suchen eine

Regression mit Bisect, nutzen eine Korrektur vor ihrem Release und bleiben mit einem Merge aktuell (Mit Git entwickeln). Eine solche Installation zu aktualisieren ist ein Verfahren in sieben Schritten mit einem Wartungsfenster, auf der Kommandozeile oder aus der Administration (Mit Git aktualisieren).

Bekannte Lücken

Die Regeln für Merge und Push sind Richtlinien: `main` hat keine Branch-Schutzregel, und Commits und Tags werden nicht signiert.

Verwandte Seiten

- Beiträge an das Projekt
- Commit-Nachrichten
- Ein Team mit Git führen
- 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.