Git-Programm und Version prüfen

Bevor PhpStorm sinnvoll mit Git zusammenarbeiten kann, muss die IDE wissen, *welches* Git-Programm auf deinem System verwendet werden soll – und ob dessen Version aktuell genug ist. Gerade unter Windows 11, wo häufig mehrere Git-Installationen parallel existieren (Git for Windows, WSL-Git, in PhpStorm gebündelte Tools), ist diese Prüfung kein optionaler Schritt, sondern die Grundlage für alle weiteren Kapitel.

Warum die Version überhaupt eine Rolle spielt

Git entwickelt sich kontinuierlich weiter, und einige Befehle, die in diesem Kurs verwendet werden, setzen bestimmte Mindestversionen voraus. Läuft im Hintergrund eine veraltete Git-Version, meldet PhpStorm mitunter kryptische Fehler oder blendet Funktionen im Menü einfach aus, ohne dies deutlich zu kommentieren.

Funktion	Benötigte Git-Version (mindestens)
`git switch` / `git restore`	2.23
`--force-with-lease` als Standardverhalten stabil	2.30+
Sparse Checkout im Cone-Modus	2.25
Partielle Klone (`--filter`)	2.19
Bessere Rebase-Autostash-Optionen	2.9+

Empfehlung: Verwende durchgängig eine aktuelle Version aus der 2.4x-Reihe oder neuer, damit sämtliche in diesem Buch beschriebenen Befehle und PhpStorm-Funktionen zuverlässig zur Verfügung stehen.

Die Git-Version über die Kommandozeile prüfen

Der schnellste und zuverlässigste Weg führt unabhängig von der IDE über ein Terminal:

```
git --version
```

Die Ausgabe sollte in etwa so aussehen:

```
git version 2.46.0.windows.1
```

Der Zusatz `windows` zeigt, dass tatsächlich *Git for Windows* aufgerufen wurde – und nicht etwa eine über WSL installierte Version oder ein Relikt einer alten Installation. Solltest du stattdessen eine deutlich ältere Version oder eine Fehlermeldung wie `git is not recognized` erhalten, liegt entweder ein PATH-Problem vor (siehe Kapitel 2) oder Git wurde noch nicht installiert.

Den Git-Pfad in PhpStorm kontrollieren

PhpStorm besitzt eine eigene Einstellung, über die festgelegt wird, welche Git-Executable die IDE tatsächlich verwendet. Diese kann von der Version abweichen, die in deiner Kommandozeile aktiv ist – etwa wenn mehrere Installationen vorhanden sind.

1. Öffne **Datei → Einstellungen** (bzw. **File → Settings** bei englischer Oberfläche).
2. Navigiere zu **Version Control → Git**.
3. Prüfe das Feld **Path to Git executable**.

Im Normalfall trägt PhpStorm hier automatisch den Pfad zur global installierten Git-Version ein, üblicherweise:

```
C:\Program Files\Git\cmd\git.exe
```

Die Test-Schaltfläche nutzen

Neben dem Pfadfeld befindet sich die Schaltfläche **Test**. Ein Klick darauf führt intern denselben Befehl aus wie `git --version` auf der Kommandozeile und zeigt das Ergebnis direkt in einem Dialogfenster an:

```
Git version 2.46.0 detected
```

Erscheint stattdessen eine Fehlermeldung, deutet dies meist auf eines der folgenden Probleme hin:

- **Falscher oder veralteter Pfad** – Git wurde neu installiert oder verschoben, PhpStorm zeigt aber noch auf den alten Speicherort.
- **Beschädigte Installation** – die `.exe`-Datei existiert, ist aber nicht ausführbar oder unvollständig.
- **Mehrere konkurrierende Installationen** – etwa eine ältere Version aus einem Paketmanager, die zufällig zuerst im Pfad gefunden wurde.

Den Pfad manuell festlegen

Sollte die automatische Erkennung fehlschlagen oder du bewusst eine andere Git-Installation verwenden wollen, lässt sich der Pfad manuell setzen:

1. Klicke im Feld **Path to Git executable** auf das Ordnersymbol.
2. Navigiere zum gewünschten `git.exe`, typischerweise unter `C:\Program Files\Git\cmd\`.
3. Bestätige die Auswahl und klicke erneut auf **Test**, um die Erkennung zu verifizieren.

Hinweis: Verwende möglichst den Pfad im `cmd`-Unterordner (`Git\cmd\git.exe`) und nicht den im `bin`-Unterordner, da Letzterer für die Nutzung innerhalb von Bash-Umgebungen optimiert ist und in Kombination mit PhpStorms interner Prozessausführung gelegentlich zu Inkonsistenzen führen kann.

Mehrere Git-Installationen unter Windows 11 erkennen

Es ist unter Windows keine Seltenheit, mehrere Git-Varianten gleichzeitig vorzufinden:

- **Git for Windows** – die im Kurs empfohlene, eigenständige Installation.
- **Git innerhalb von WSL** (Windows Subsystem for Linux) – separat installiert, mit eigenem Pfad und eigener Konfiguration.
- **Git aus Paketmanagern** wie Chocolatey oder Scoop – funktioniert meist zuverlässig, kann aber zu Versionskonflikten führen, wenn parallel weitere Installationen existieren.

PhpStorm sollte ausschließlich auf die für dieses Buch vorgesehene Git-for-Windows-Installation verweisen. Solltest du zusätzlich mit WSL arbeiten, achte darauf, Projekte nicht unbeabsichtigt zwischen beiden Welten zu vermischen, da sich Zeilenenden, Dateirechte und Pfadformate unterscheiden können.

Automatische Erkennung vs. explizite Konfiguration

Für die meisten Einzelplatz-Setups reicht die automatische Erkennung vollkommen aus. In Team- oder Firmenumgebungen empfiehlt sich dennoch, den Pfad einmal bewusst zu prüfen und zu dokumentieren – besonders dann, wenn:

- Projekte über mehrere Rechner mit unterschiedlichen Installationswegen hinweg geteilt werden,

- Continuous-Integration-Umgebungen (siehe Kapitel 20) eine bestimmte Git-Version voraussetzen,
- ältere Legacy-Systeme im Unternehmen noch mit veralteten Git-Versionen arbeiten.

Git aktualisieren

Wird eine veraltete Version erkannt, lässt sich Git for Windows unkompliziert aktualisieren:

```
git update-git-for-windows
```

Dieser Befehl prüft auf eine neuere Version und führt bei Bedarf den Installer automatisch aus. Alternativ kann die aktuelle Version jederzeit manuell von der offiziellen Git-for-Windows-Seite heruntergeladen und installiert werden – bestehende Konfigurationswerte (siehe Kapitel 2) bleiben dabei in der Regel erhalten.

Checkliste für diesen Schritt

Bevor du mit den nächsten Abschnitten dieses Kapitels fortfährst, sollten folgende Punkte erfüllt sein:

- `git --version` liefert in der Kommandozeile eine aktuelle Version (mindestens 2.4x).
- PhpStorm zeigt unter **Version Control → Git** denselben Pfad und dieselbe Version an.
- Die **Test**-Schaltfläche bestätigt die Erkennung ohne Fehlermeldung.
- Es ist klar, welche Git-Installation tatsächlich aktiv ist, falls mehrere auf dem System vorhanden sind.

Ist all dies sichergestellt, steht einer reibungsvollen Integration von Git in PhpStorm nichts mehr im Weg – und die folgenden Abschnitte zur Versionskontrolle eines Projekts können ohne Umwege beginnen.

Revision #1
Created 2026-07-26 17:04:50 UTC by art10m
Updated 2026-07-26 17:05:51 UTC by art10m