Feature-Branch in den Main-Branch mergen – Schritt für Schritt in PhpStorm □□

Du hast in deinem Feature-Branch (z. B. `login-formular`) alle Änderungen abgeschlossen und getestet. Jetzt ist es Zeit, diese Arbeit in deinen Hauptbranch (meist `main` oder `master`) zu integrieren. Diesen Vorgang nennt man **Merge** – und PhpStorm macht ihn dir besonders komfortabel.

Das Grundprinzip verstehen

Beim Mergen passiert Folgendes: Git nimmt alle Änderungen aus deinem Feature-Branch und **fügt sie in einen anderen Branch ein** (den Zielbranch). Wichtig dabei ist die Reihenfolge:

```
flowchart LR
    A["1. Wechsle zum Zielbranch\n(z.B. main)"] --> B["2. Merge den Feature-Branch\nin den aktuellen Branch"]
    B --> C["3. Aenderungen sind\njetzt in main"]
```

> □□ **Merke:** Du wechselst immer **zuerst** in den Branch, der die Änderungen **empfangen** soll, und holst dann den anderen Branch hinein.

Schritt-für-Schritt-Anleitung

1. Offene Änderungen committen

Bevor du mergst, stelle sicher, dass in **beiden Branches** keine uncommitteten Änderungen herumliegen. Öffne das Commit-Fenster (`Ctrl + K` / `Cmd + K`) und prüfe, ob alles sauber committet ist. Falls du noch Änderungen im Feature-Branch hast, committe diese zuerst.

2. Zum Main-Branch wechseln

Da du die Änderungen **in den Main-Branch** integrieren willst, musst du dorthin wechseln:

1. Schau in die **rechte untere Ecke** von PhpStorm – dort siehst du den aktuellen Branch-Namen.
2. **Klicke darauf** – es öffnet sich das Branch-Popup.
3. Suche deinen Main-Branch (z. B. `main` oder `master`) unter **Local Branches**.
4. Klicke auf den Branch-Namen und wähle **Checkout**.

PhpStorm wechselt nun zum Main-Branch. Du siehst, dass sich der Name in der Statusleiste ändert, und dein Working Directory zeigt jetzt den Stand des Main-Branches.

3. Den Feature-Branch mergen

Jetzt kommt der eigentliche Merge:

1. Klicke erneut auf den **Branch-Namen** in der Statusleiste (jetzt steht dort „main").
2. Im Popup siehst du unter **Local Branches** deinen Feature-Branch (z. B. `login-formular`).
3. **Klicke auf den Feature-Branch** und wähle im Untermenü **Merge into Current**.

PhpStorm führt nun den Merge durch. Wenn alles glatt läuft (keine Konflikte), siehst du eine kurze Erfolgsmeldung, und die Änderungen aus deinem Feature-Branch sind jetzt Teil des Main-Branches.

4. Das Ergebnis überprüfen

Nach dem Merge lohnt sich ein kurzer Blick ins Git-Log (`Alt + 9` oder **Git → Show Git Log**):

- Du siehst einen neuen **Merge-Commit**, der die beiden Entwicklungslinien zusammenführt.
- Die Commit-Historie zeigt, dass die Commits aus dem Feature-Branch nun auch im Main-Branch enthalten sind.

Was passiert im Hintergrund?

Wenn du in PhpStorm „Merge into Current" auswählst, führt Git im Hintergrund folgenden Befehl aus:

```
git merge login-formular
```

Dabei erstellt Git (je nach Situation) entweder:

- Einen **Merge-Commit**, der beide Branches zusammenführt und als „Knotenpunkt" in der Historie sichtbar ist.
- Einen **Fast-Forward-Merge**, wenn der Main-Branch seit der Erstellung des Feature-Branches keine eigenen Änderungen hatte. In diesem Fall wird kein extra Merge-Commit erstellt – die Commits werden einfach „vorgeschoben".

Was tun bei Merge-Konflikten? ⚠

Manchmal hat Git ein Problem: Wenn **dieselbe Stelle** in einer Datei in beiden Branches unterschiedlich geändert wurde, kann Git nicht automatisch entscheiden, welche Version gelten soll. Das nennt man einen **Merge-Konflikt**.

In diesem Fall öffnet PhpStorm automatisch ein **Konfliktlösungs-Fenster**:

1. Du siehst **drei Spalten**: Links dein aktueller Branch (Main), rechts der Feature-Branch, in der Mitte das Ergebnis.
2. Für jede Konfliktstelle kannst du entscheiden:
   - **Accept Left** – die Version aus Main übernehmen
   - **Accept Right** – die Version aus dem Feature-Branch übernehmen
   - **Manuell bearbeiten** – in der Mitte selbst eine Kombination schreiben
3. Wenn alle Konflikte gelöst sind, klicke auf **Apply**.
4. PhpStorm erstellt dann automatisch den Merge-Commit.

> □□ **Tipp:** Konflikte klingen schlimmer als sie sind. In den meisten Fällen ist schnell klar, welche Version die richtige ist – und das visuelle Tool von PhpStorm macht die Lösung sehr intuitiv.

Nach dem Merge: Feature-Branch löschen?

Wenn der Merge erfolgreich war und du den Feature-Branch nicht mehr brauchst, kannst du ihn aufräumen:

1. Klicke wieder auf den **Branch-Namen** in der Statusleiste.
2. Finde deinen Feature-Branch unter **Local Branches**.
3. Klicke darauf und wähle **Delete**.

Das hält deine Branch-Liste übersichtlich. Die Commits gehen dabei **nicht verloren** – sie sind jetzt Teil des Main-Branches und bleiben in der Historie erhalten.

Zusammenfassung

Schritt	Aktion in PhpStorm
1. Alles committen	`Ctrl/Cmd + K` → Commit
2. Zum Zielbranch wechseln	Branch-Popup → Main → **Checkout**
3. Feature-Branch mergen	Branch-Popup → Feature-Branch → **Merge into Current**
4. Bei Konflikten	Konflikt-Dialog nutzen → **Apply**
5. Optional aufräumen	Branch-Popup → Feature-Branch → **Delete**

Mit dieser Routine kannst du sicher neue Features entwickeln, testen und dann sauber in deinen Hauptbranch integrieren – genau so, wie es in professionellen Projekten üblich ist. □□

Revision #1
Created 2026-06-17 01:35:21 UTC by art10m
Updated 2026-06-17 01:40:33 UTC by art10m