Allgemein

Diese Seite gibt einen Überblick darüber, wie wir in der SQUEEZE-Entwicklung mit Git arbeiten: welche Branches es gibt, wofür sie da sind und wie Änderungen ihren Weg in ein Release finden.

Der Umfang unserer Git-Projekte ist unterschiedlich, deshalb gelten nicht überall die gleichen Anforderungen – je kleiner das Projekt, desto einfacher und weniger strikt der Workflow. Der hier beschriebene Workflow gilt für **SQUEEZE und das DEXP-Frontend**.

Das Repository liegt in Azure DevOps im Projekt *DEXPRO Platform*:

```
git clone git@ssh.dev.azure.com:v3/DEXPRO/DEXPRO%20Platform/SQUEEZE
```

Branch-Modell

Wir arbeiten mit einem langlebigen Entwicklungs-Branch und pro Minor-Version einem Release-Branch:

Branch	Bedeutung
`develop`	Aktueller Entwicklungsstand und Ausgangspunkt für neue Arbeit. Der Branch *soll* möglichst immer lauffähig sein.
`release/<major.minor>` (z. B. `release/2.34`)	Enthält die getaggten Releases einer Minor-Version. Jeder getaggte Commit *muss* lauffähig sein. Hier entstehen Patch-Releases und Hotfixes. Release-Branches werden **nicht** pauschal in den `develop` zurückgemerged.
Themen-Branches (`feature/`, `task/`, `fix/`, `bugfix/`, `hotfix/`)	Kurzlebige Branches für eine konkrete Änderung. Sie werden per Pull Request in `develop` oder einen `release/`-Branch gemerged und danach gelöscht.

Namenskonvention für Branches

Branch-Namen werden auf Englisch formuliert und beginnen mit der Kategorie, gefolgt von der Work-Item-/Ticketnummer und einer kurzen Beschreibung:

```
feature/18157-image-tools-refactor
task/17609-add-savon-cli-tool
fix/17470-log-fix
```

Ist ein Branch gezielt für eine bestimmte Version gedacht, wird die Version mit aufgenommen – so ist auf einen Blick erkennbar, wohin die Änderung gehört:

```
hotfix/2.33-18092-createXMLErrorPDF-generates-unique-pdf
fix/2.32/kosit-pdf-creation-crash
```

Commits

- Commit-Messages werden auf Englisch formuliert.
- Commits möglichst so schneiden, dass eine zusammenhängende Änderung in einem Commit liegt. Das ist nicht nur für das Review angenehmer, sondern vor allem die Grundlage dafür, dass ein Fix später sauber in einen anderen Branch ge-cherrypickt werden kann.

Änderungen einbringen

Änderungen kommen grundsätzlich über einen **Pull Request** in `develop` oder einen `release/`-Branch – kein direkter Push auf diese Branches. Der PR ist der Ort für Code-Review, Build- und Testlauf.

Der typische Ablauf für ein neues Feature oder eine Task:

1. Aktuellen `develop` auschecken und aktualisieren.
2. Themen-Branch nach obiger Konvention anlegen.
3. Entwickeln, committen, Branch pushen.
4. Pull Request nach `develop` erstellen und mit dem Work Item verknüpfen.
5. Nach Review und grünem Build mergen; der Branch wird anschließend gelöscht.

Bugfixing über mehrere Versionen

Beim Bugfixing kommt neben der Problemanalyse die Frage dazu, **auf welchem Produktstand** der Bug auftritt und in welchen Versionen er behoben werden muss. Im Idealfall wird der verursachende Commit identifiziert – daraus ergibt sich, welche Fixes wo angebracht sind.

Muss ein Fix rückwirkend in eine ausgelieferte Version, sollte wie folgt vorgegangen werden:

- **Start auf dem `release/`-Branch:**
  - Fix vom betroffenen `release/`-Branch abzweigen
  - Problem beheben und commiten
  - Per PR in den Release-Branch zurückbringen
- **Backmerge in nachfolgende Release-Branches/Develop:**

    - Danach muss der fix in alle folgenden Release-Branches bzw den Develop gemerged werden (z.B. `release/2.33` -> `release/2.34` -> `develop`)
    - Dazu wird zunächst der Ziel-Branch, auf den gemerged werden soll ausgewählt (nächster Release-Branch oder develop)
    - Davon wird ein neuer Branch erstellt (`task/XXXX-backmerge-release-xy`)
    - Der Release-Branch wird auf den neu erstellten Branch gemerged und alle Konflikte behoben.
    - Der Backmerge-Branch wird gepusht und ein PR erstellt

Auf diese Weise ist sichergestell, dass ein Bugfix, der im Release landet, auch bis auf den aktuellen develop durchgereicht wird und nicht unterwegs "verloren geht".

Weiterführend

- Code Reviewing und Pull Requests
- Versionierung
- Release und Deployment

Revision #5
Created 2026-08-18 08:20:00 UTC by Florian Vick
Updated 2026-08-18 08:32:48 UTC by Florian Vick