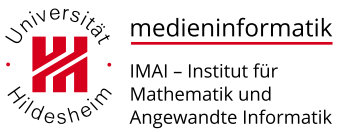
Werkzeuge

Git und (andere) Kooperationswerkzeuge

Jörg Cassens, Jens Rademacher, Bastian Stender

Medieninformatik Praktikum
SoSe 2017



Inhaltsverzeichnis

1 git: Theorie

Nutzen von und Anforderungen an Versionsverwaltungen

- Verwaltung unterschiedlicher Versionen einer Datei
- Protokollierung von Änderungen
  - Änderung,
  - Zeitpunkt,
  - Person
- Möglichkeit des Zugriffs auf vorherige Versionen
- Unterstützung mehrerer BenutzerInnen
- Unterstützung von Entwicklungszweigen, Merging, Datensicherheit

1.1 Architekturen

Architekturen (1)

- Lokale Versionsverwaltung
  - Versionierung einer einzelnen Datei mit grundlegenden Verwaltungsfunktionen (Protokollierung, Abruf, Wiederherstellung)
  - **Implementierungen**: RCS, proprietäre Software
- Erste Generation, nicht für Gruppenarbeit geeignet

Architekturen (2)

- Zentrale Versionsverwaltung
  - Einsatz eines zentralen Verwaltungsservers, Entwicklung auf Clients
  - Versionsgeschichte liegt auf dem Server
  - Rechteverwaltung auf dem Server
  - **Implementierungen**: CVS (eingestellt), SVN
- Zweite Generation, für Gruppenarbeit geeignet, auf Server angewiesen

Architekturen (3)

- Verteilte Versionsverwaltung
  - Eigenes Repository auf jedem Client
  - Kein zentraler Server notwendig
  - Abgleich der Repositories untereinander
  - Versionsgeschichte kann lokal auf jedem Client liegen
  - Ermöglicht parallele Entwicklung mit anschließendem (möglichst weitgehend automatisiertem) Merge ("Nichtlineare Entwicklung")
  - **Implementierungen**: Mercurial, Git
- Dritte Generation, für Gruppenarbeit geeignet, offline und nichtlineare Entwicklung

1.2 git

Eigenschaften

- Dezentrale Versionsverwaltung
- Entwickelt von Linus Torvalds für die Versionierung des Linux-Kernels
- Kein zentraler Server vorhanden
- Unterstützung nicht-linearer Programmierung durch Branching und Merging
- Unterstützung vieler Übertragungsprotokolle
- Revisions haben keine aufsteigende ID sondern Hash-Werte
- Authentifizierung bei Repository-Hostern häufig per Private Key

Vor- und Nachteile

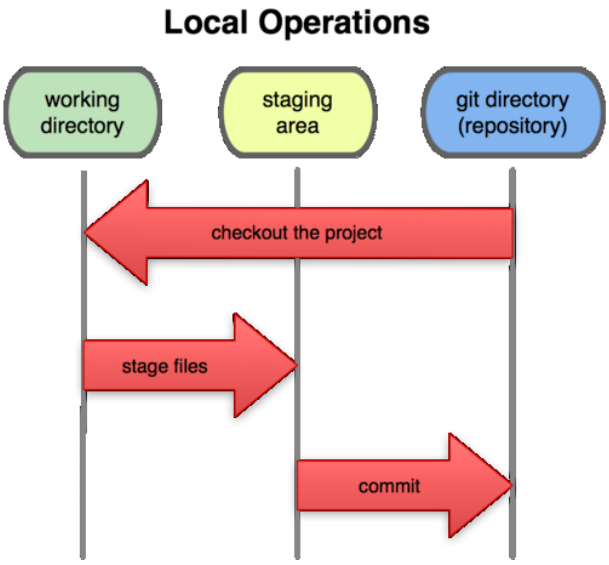
- Vorteile
  - Kein Server notwendig
  - keine verstreuten ".svn"-Ordner (lediglich ein .git-Ordner im Hauptverzeichnis)
  - Effizienter Arbeit mit dem Repository (BRANCH, DIFF, MERGE)
- Nachteile
  - Disziplin notwendig
  - Gewöhnungsbedürftiger Einsatz unter Windows
  - "Unbequeme" Lernkurve
  - Schlechte Performanz bei großen Dateien

2 git: Praxis

2.1 Repository anlegen

Initialisierung und .gitignore

- Anlegen eines neuen Repositories:
  - **git init** (lokal im Projektordner, .git-Verzeichnis)
  - **git --bare init** (örtlich entferntes Repository)
- Dateien generell von der Versionierung ausschließen:
  - Anlegen einer Textdatei mit dem Namen **.gitignore** im Hauptverzeichnis
  - Eine Regel pro Zeile (etwa: ***.pyc**)



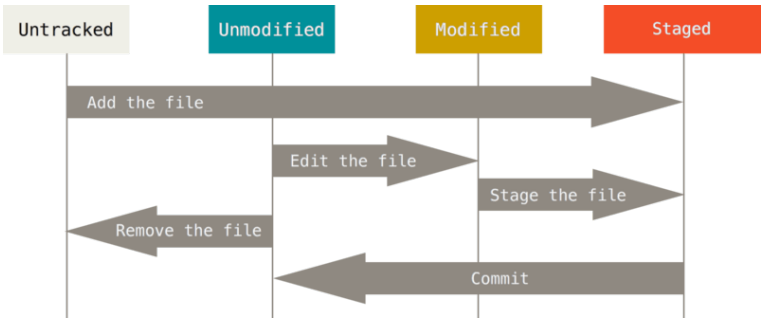
Workflow

2.2 Standard-Aufgaben

Grundfunktionen

- File-Status:
  - **Untracked** (unbeobachtet, nicht Teil der Versionsverwaltung)
  - **Modified** (verändert gegenüber letztem Commit)
  - **Staged** (für den Commit vorgemerkt)

  - **Unmodified** (entspricht dem letzten „Commited“)
- Status auslesen:
  - **git status**
- Hinzufügen oder Stagen von Dateien:
  - **git add FILE(S)**
- Commit:
  - **git commit -m “Zusammenfassung“**
- Stage (deleted & modified) & Commit:
  - **git commit -a -m “Zusammenfassung“**
- Unterschiede zu HEAD anzeigen:
  - **git diff HEAD**



lifecycle

Änderungen Zurücknehmen

- Änderungen des letzten Commits rückgängig machen, Commit bleibt erhalten
  - **git revert**
- HEAD-pointer auf genannten Commit setzen
  - **git revert COMMIT**
- Stage/Arbeitsverzeichnis auf den Stand des letzten Commits setzen
  - **git reset**

2.3 Remote-Repositories

Remote-Repositories

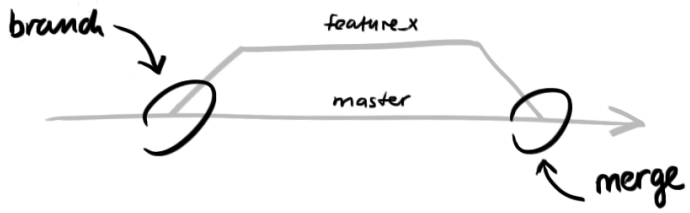
- Lokale Kopie eines entfernten Repositories erstellen:
  - **git clone URL**
- Entferntes Repository hinzufügen:
  - **git remote add REMOTENAME URL**
- Lokale Änderungen in entferntes Repository übertragen:
  - **git push REMOTENAME BRANCH**
  - REMOTENAME ist häufig **origin**
  - Standard-Branch ist **master**

2.4 Branching

Branching

- Branch erstellen
  - **git branch NAME**
- Alle Branches anzeigen
  - **git branch -a**
- In Branch wechseln
  - **git checkout NAME**
- Branch mit Master mergen
  - **git checkout master** (in den Master-Zweig wechseln)
  - **git merge NAME** (Mergen)
- Branch löschen
  - **git branch -d NAME**

Branches



Arbeiten mit Branches

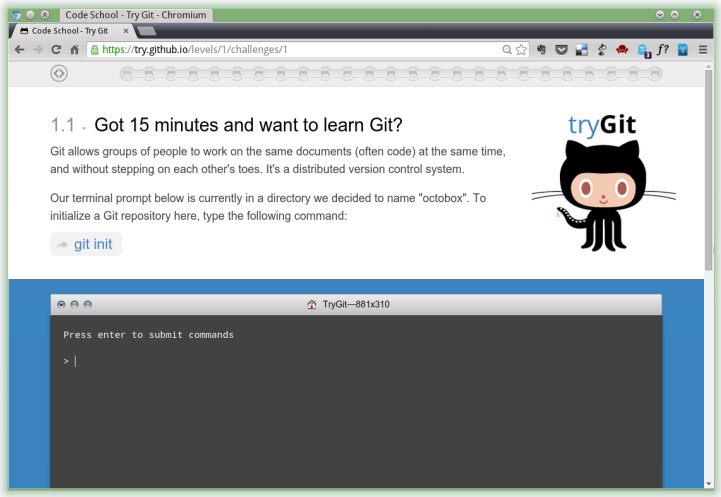
2.5 Sonstiges

Einstellungen und Hilfe

- (Globale) Einstellungen festlegen:
  - **git config [--global] user.name "John Doe"**
  - **git config [--global] user.email john@example.com**
  - **git config [--global] core.autocrlf input** (Linux)
- Hilfe
  - **git COMMAND --help**

2.6 Tutorial

github Tutorial

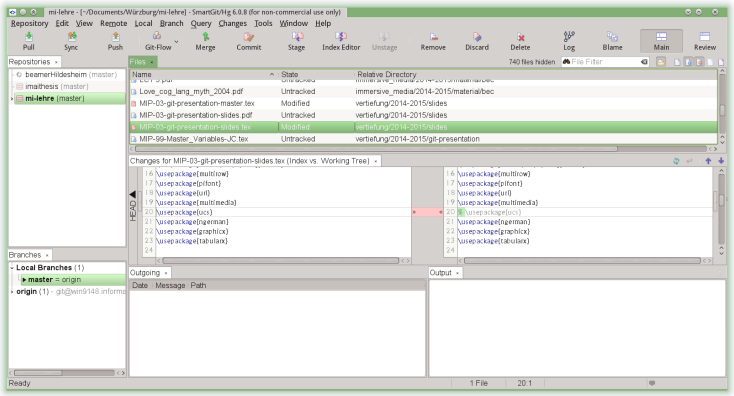


☞ try.github.com

3 git: Tools

3.1 git: GUI

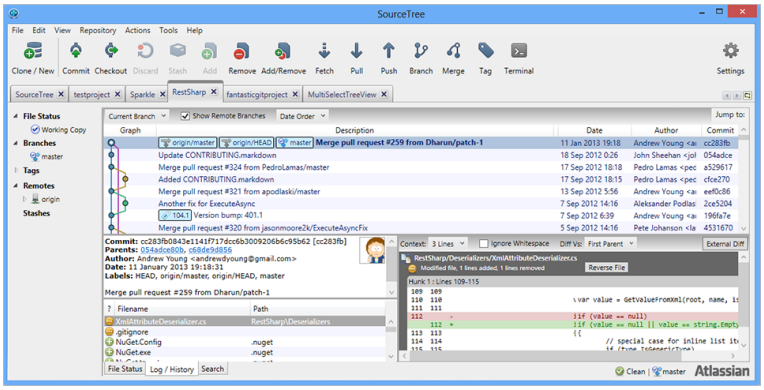
smartgit



☞ www.syntevo.com/smartgit

Cross-Platform (Linux, Mac, Windows), free for non-commercial use

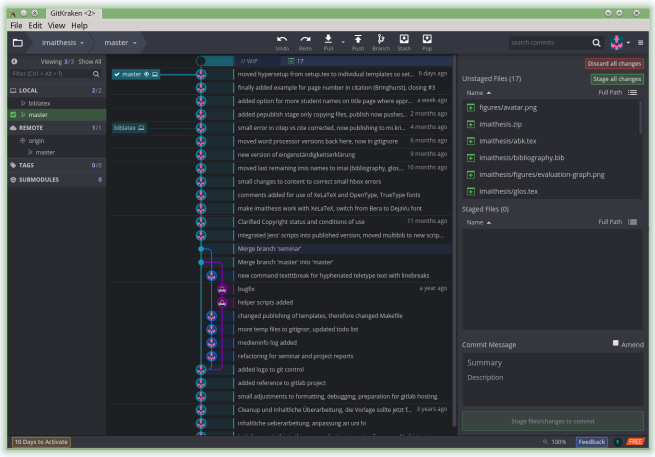
SourceTree



☞ SourceTreeApp.com

Mac, Windows; free to use, registration required

GitKraken



☞ gitkraken.com

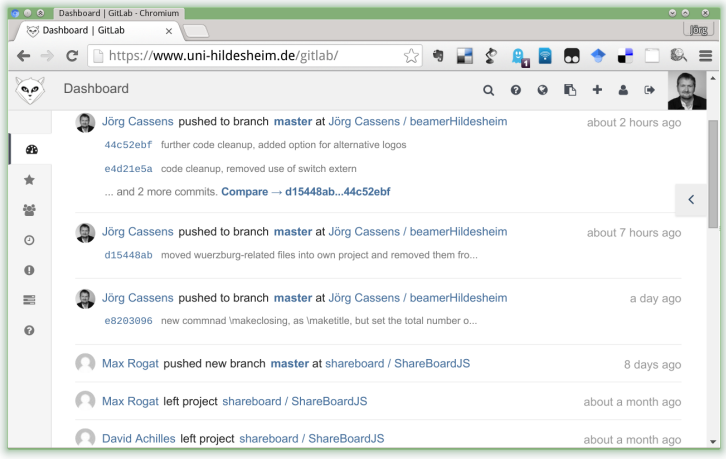
Cross-Platform (Linux, Mac, Windows), free for non-commercial use, registration required

3.2 git: Project Hosting

Project Hosting

- Hilfen für Projekte durch:
  - Issuetracker
  - Wiki (Markdown)
  - Statistiken (Gamification)
  - Download von Projekten
  - Releases
- Gruppenarbeit
- Forks und Pull-Requests
  - Möglichkeit, sich einfach in neue Projekte einzubringen
  - “Standing on the shoulders of giants”
- Mehrere Services mit unterschiedlichen Vor- und Nachteilen
  - gitlab
  - github
  - bitbucket

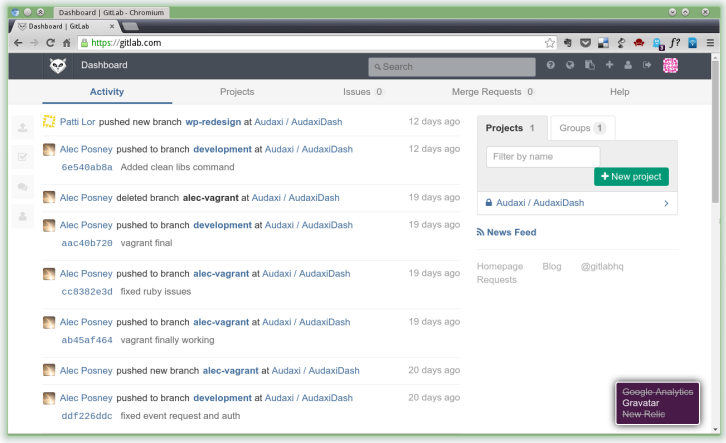
gitlab (Uni Hildesheim)



☞ www.uni-hildesheim.de/gitlab

Hosted OSS-System

gitlab (Kommerziell)

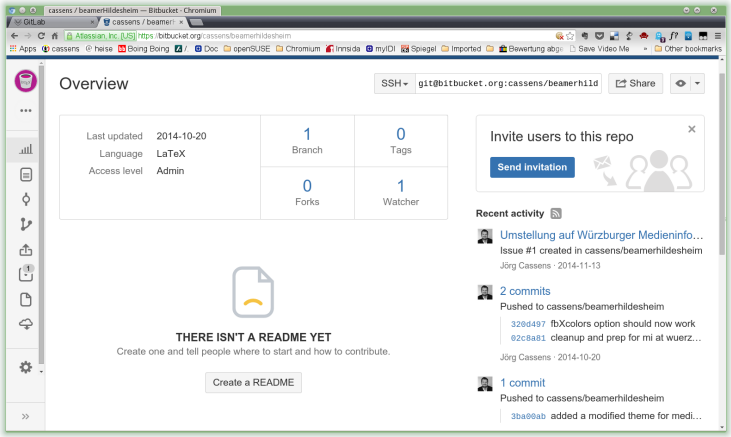


☞ gitlab.com
Kommerzielle Variante, hosted

gitlab (OSS)

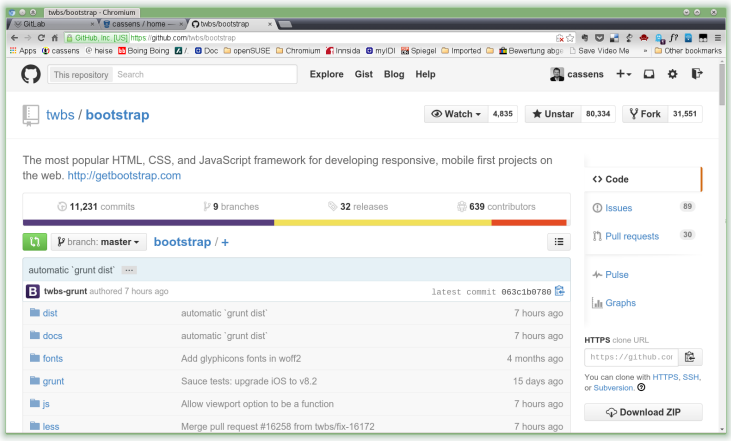
☞ about.gitlab.com
Self-Hosted OSS-System

Atlassian Bitbucket



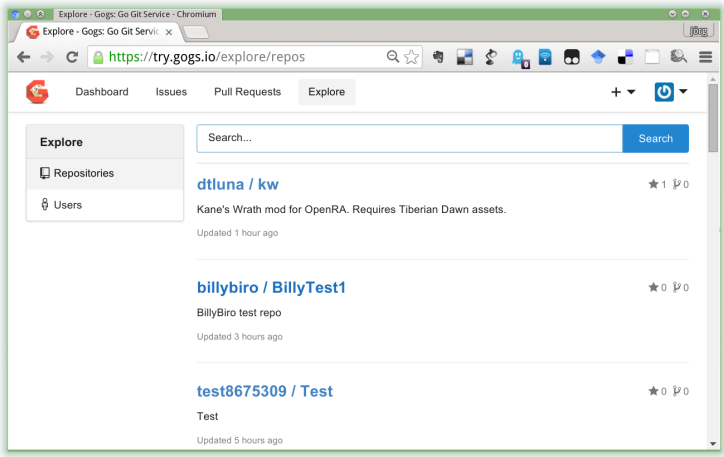
☞ bitbucket.org
Kommerziell, hosted, Freemium

github



☞ www.github.com
Kommerziell, hosted, Freemium

gogs



☞ gogs.io
Self-Hosted OSS-System

Verwendung im Praktikum

- Die einzelnen Gruppen sollen jeweils ein Projekt bei einem der genannten Services anlegen
  - ☞ uni-hildesheim.de/gitlab – Universität Hildesheim
  - ☞ www.gitlab.com kostenlose öffentliche und private Projekte
  - ☞ www.bitbucket.com kostenlose öffentliche und private Projekte, kostenlos nur eingeschränkte Teamgrößen
  - ☞ www.github.com kostenlos nur für öffentliche Projekte
- Zu diesem Projekt werde ich eingeladen
  - Zugriff auf Code und Dokumentation
  - Möglichkeit, Tickets zu erstellen
- Ich empfehle das gitlab der Uni
- Wer das nicht benutzen möchte (Zugriff, Uptime) sollte eines der anderen Systeme nutzen

4 Projektmanagement

Bedarf

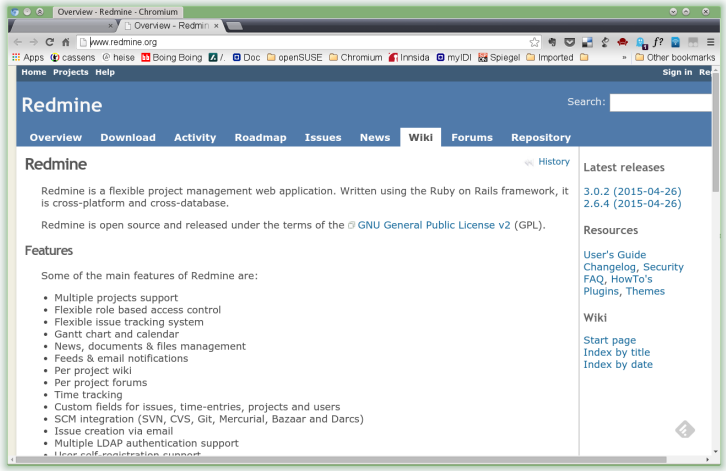
- Versionsverwaltung
  - Wie gesehen
- Ticketing
  - Bei den genannten Hosting-Providern im Grundsatz enthalten
- Projektplanung
  - Zumindest: Meilensteine und Ticketing
- Dokumentation
  - Wiki bei den genannten Providern
  - LaTeX im git
  - Kollaborative Editoren
- Kommunikation & Koordination
  - Mehr als facebook und Dropbox
- Automatisierung
  - Wenn im git-repo etwas passiert, können auch andere Dinge angestoßen werden

4.1 Ticketing und Projektplanung

Tickets und Meilensteine

- Die git-hoster stellen in der Regel ein eigenes Ticket-System zur Verfügung
- Zusammen mit Meilensteinen ist rudimentäre Projektplanung möglich
- Vorteile
  - Arbeit mit vorhandenen Werkzeugen
- Nachteile
  - In der Regel nicht sehr flexibel

Redmine

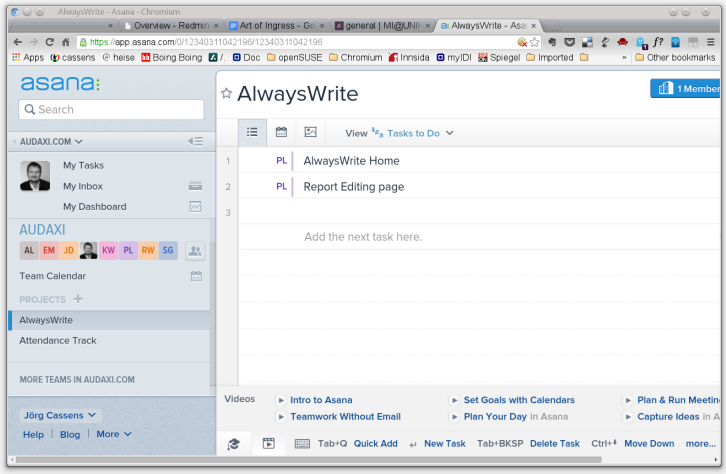


☞ www.redmine.org
integriertes System

Redmine

- Umfangreiches integriertes System
  - Projektmanagement
  - Zeiterfassung
  - Dokumentation
  - git-Repositories
- Vorteile
  - Viele Möglichkeiten, erweiterbar
- Nachteile
  - Kein reiner Projektplaner
  - Self-Hosting
- Alternative
  - ☞ trac

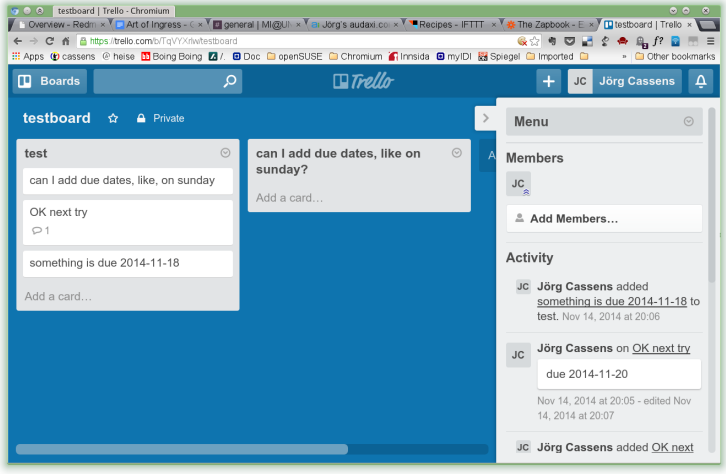
asana



☞ asana.com

Task Management, Freemium

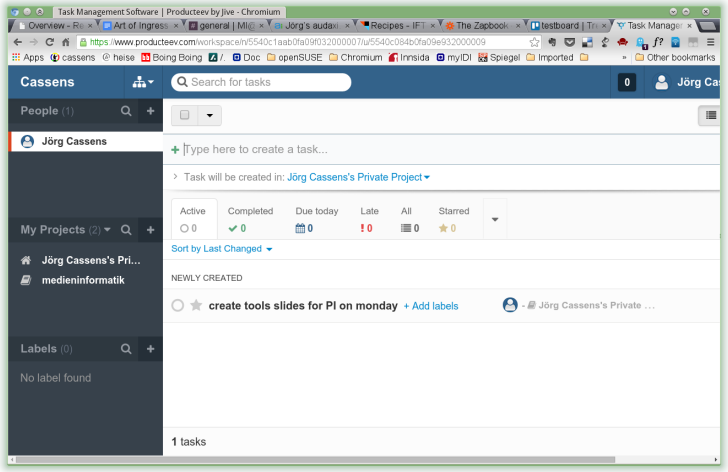
Trello



☞ trello.com

Kanban-Style Task Management, Freemium

producteev



☞ www.producteev.com
Task Management, Freemium

asana, Trello, producteev

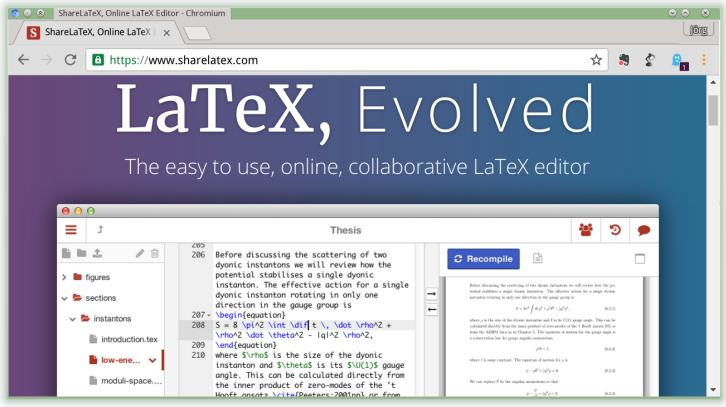
- Fokus auf Projektmanagement
- Unterschiedliche Philosophien (kanban, "traditionell")
- Vorteile
  - Mächtigkeit
- Nachteile
  - Mächtigkeit

4.2 Dokumentation

LaTeX

- git funktioniert sehr gut bei Textdateien
- Für Binary Blobs wie PDF oder Dateien von Textverarbeitungen eher ungeeignet
- LaTeX ist textbasiert
  - Dokumentation in einem doc-repo
- Nachteile
  - Assets wie Bilder werden semi-optimal behandelt
  - Änderungsverfolgung im Text
    - mit etwas Handarbeit mit latexdiff realisierbar
- Vorteile
  - Professionelle Satzqualität
  - Handhabung mit den gleichen Werkzeugen

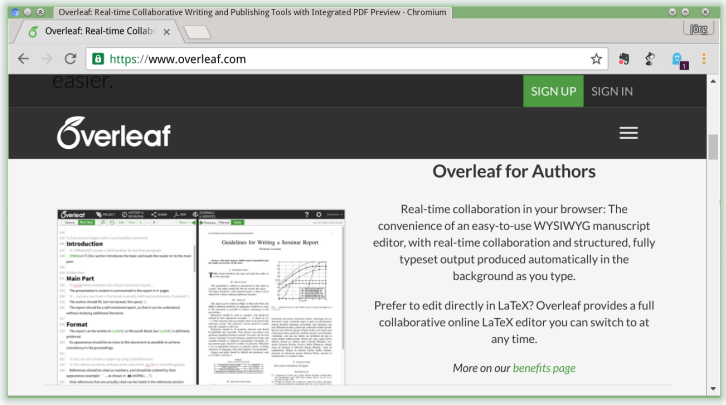
ShareLaTeX



☞ sharelatex.com

Collaborative (nur in bezahlter Version), online LaTeX-shell, Freemium

overleaf



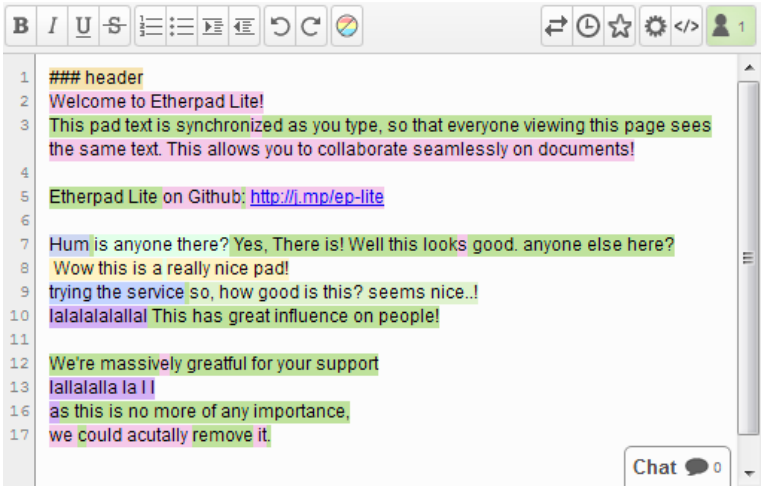
☞ overleaf.com

Collaborative, online LaTeX-shell, Freemium

Wiki

- "Mitgelieferte" Wikis verwenden
- Vorteile
  - Einfache Markdown-Syntax
  - Online
  - Gleiche Werkzeuge
- Nachteile
  - Einfache Markdown-Syntax
  - Online
- Aus Markdown lassen sich mit verschiedenen Werkzeugen andere Formate erzeugen (HTML, PDF, LaTeX, ODF)
  - Beispiel pandoc, ☞ johnmacfarlane.net/pandoc
  - Beispiel MultiMarkdown, ☞ fletcherpenney.net/multimarkdown

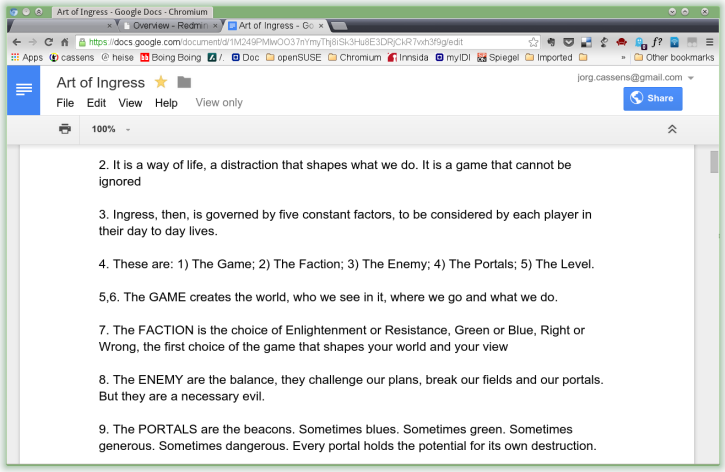
Etherpad



☞ etherpad.org, ☞ epad.hosting.uni-hildesheim.de

OSS Collaborative online Text Editor, hosted oder self-hosted

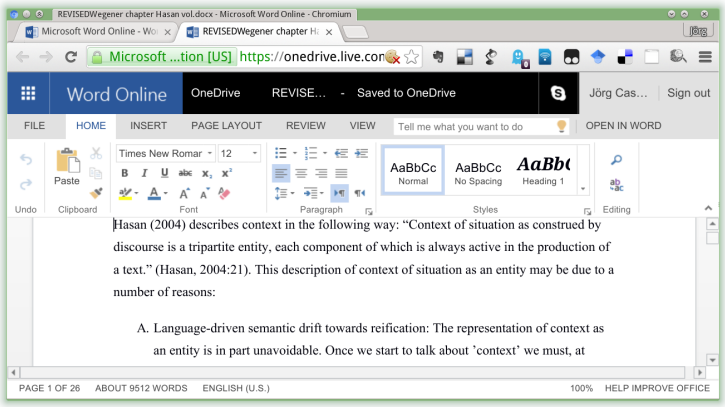
Google Docs



☞ docs.google.com

Collaborative online Word Processor

Microsoft Word Online

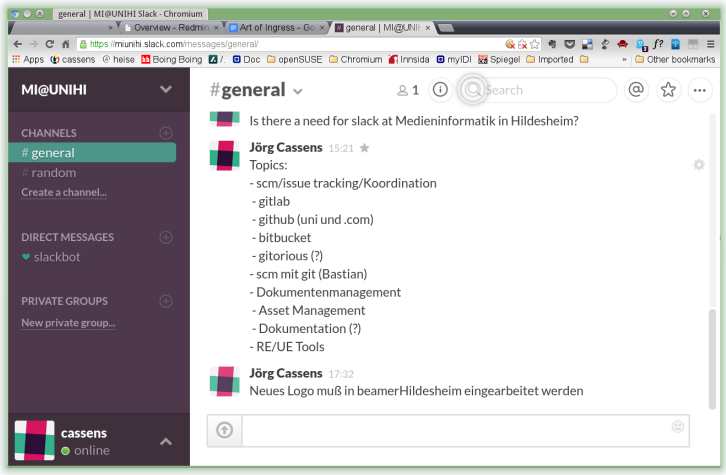


☞ office.live.com

Collaborative online Word Processor

4.3 Kommunikation & Koordination

slack

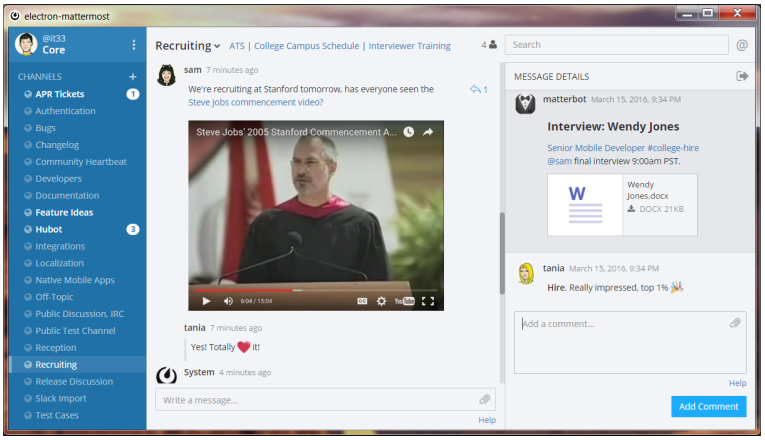


☞ slack.com
Kommerziell, Freemium

slack

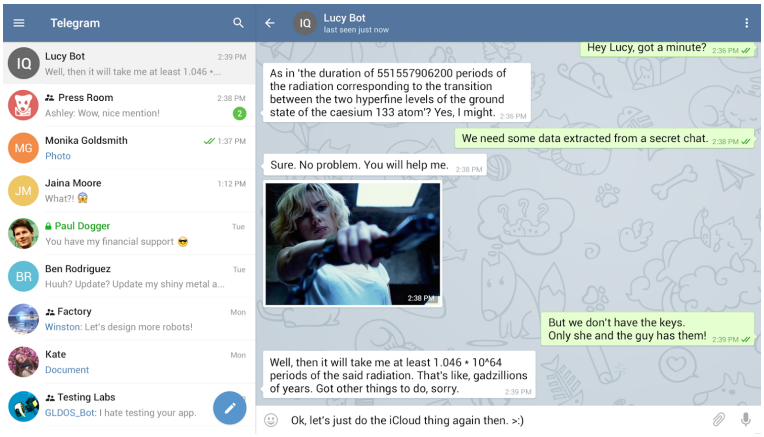
- Kommunikation in Teams in sogenannten Kanälen
- Strukturierter als Hangouts, übersichtlicher als facebook, einfacher als IRC
- Vorteile
  - Diverse Funktionen und Hooks
  - Gute Verbindung mit anderen Systemen (git commit-messages)
  - Kaum Einschränkungen für kostenlose Accounts
- Nachteile
  - ...

Mattermost



☞ mattermost.org
Self-hosted Slack Clone, bei gitlab "dabei"

Telegram



☞ telegram.org

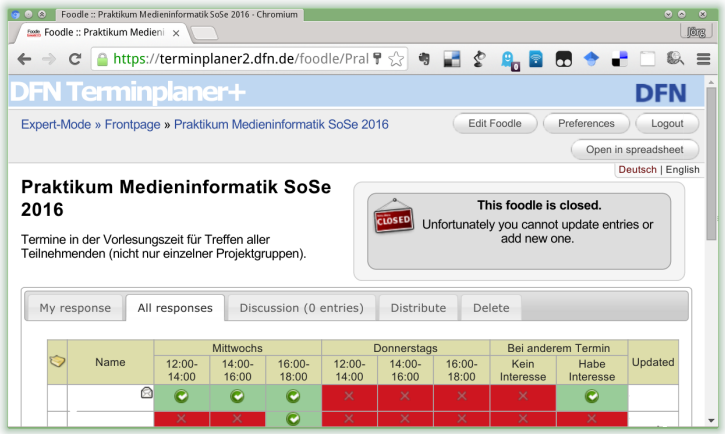
Messenger, End-to-End-Encryption bei privaten chats, Gruppen

Signal

☞ whispersystems.org

Messenger, End-to-End-Encryption, Desktop-Version, Gruppen

Foodle



☞ terminplaner2.dfn.de

Terminplaner, DFN

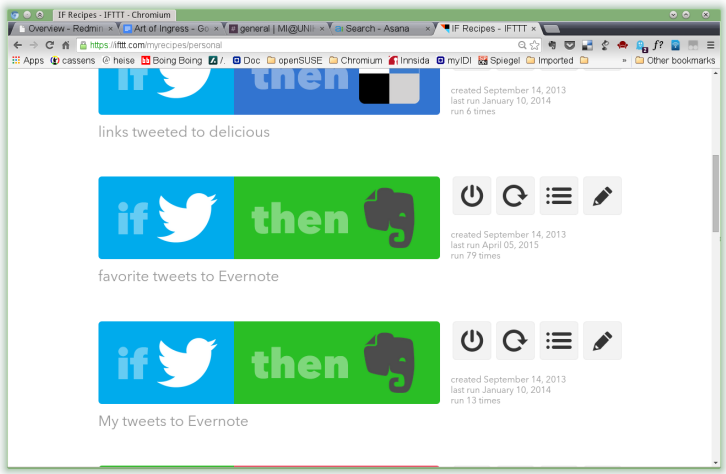
4.4 Automatisierung

Hooks

- Git stellt sogenannte Hooks zur Verfügung, mit denen Aktionen angestoßen werden können

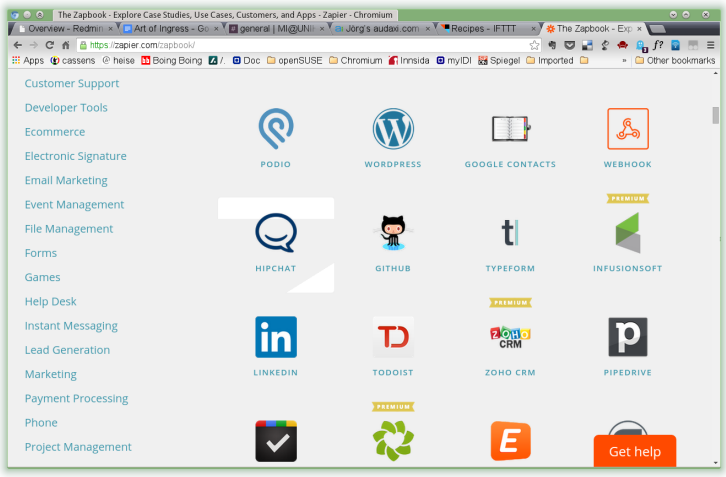
- Diese sind teilweise bei den genannten Providern vorhanden
  - Mail bei Commit
  - Slack-Messages bei Commit

if this then that



☞ ifttt.com
If this then that, Kommerziell, Freemium

zapier



☞ zapier.com
Ähnliches Konzept wie ifttt, Kommerziell, Freemium

ifttt, zapier

- Ermöglichen es, unterschiedliche Datenquellen und Datensenken miteinander zu verknüpfen
- ifttt offener in der Nutzung, aber zapier hat manchmal bessere Anbindungen
- Vorteile
  - Automatisierung
- Nachteile
  - Man gibt einem externen Dienstleister Zugriff auf viele unterschiedliche Systeme

4.5 Empfehlung

Empfehlung

- Source-Code Management (git)
- Dokumentation von Anfang an (wiki, LaTeX)
- Tickets
- Meilensteine
- Chat (slack, Telegram)

Schluß

git: Info & Tools

- Basis
  - ☞ git-scm.com – Git für Windows (Installieren und deutsche Sprachdatei in .old umbenennen)
- Tutorial & Dokumentation
  - ☞ try.github.com – Github-Tutorial mit der Octocat
  - ☞ git-scm.com/book/de – Deutsches Git-Buch
- GUI-Werkzeuge
  - ☞ www.syntevo.com/smartgit
  - ☞ SourceTreeApp.com
  - ☞ gitkraken.com

git: Hosting

- Hosted
  - ☞ uni-hildesheim.de/gitlab – Universität Hildesheim
  - ☞ www.gitlab.com kostenlose öffentliche und private Projekte
  - ☞ www.bitbucket.com kostenlose öffentliche und private Projekte, kostenlos nur eingeschränkte Teamgrößen
  - ☞ www.github.com kostenlos nur für öffentliche Projekte
- Self-Hosted
  - ☞ about.gitlab.com – self-hosted
  - ☞ gogs.io – self-hosted

Projektmanagement & Dokumentation

- Projektmanagement
  - ☞ www.redmine.org
  - ☞ trac.edgewall.org
  - ☞ asana.com
  - ☞ trello.com
  - ☞ www.producteev.com
- Dokumentation
  - LaTeX & git
  - ☞ sharelatex.com
  - ☞ overleaf.com
  - ☞ etherpad.org

  - ☞ epad.hosting.uni-hildesheim.de
  - ☞ docs.google.com
  - ☞ office.live.com
  - ☞ johnmacfarlane.net/pandoc
  - ☞ fletcherpenney.net/multimarkdown

Kommunikation, Koordination, Automatisierung

- Kommunikation & Koordination
  - ☞ slack.com – Slack
  - ☞ mattermost.org – Slack-Clone
  - ☞ telegram.org – telegram
  - ☞ whispersystems.org – Signal
  - ☞ terminplaner2.dfn.de – Foodle
- Automatisierung
  - ☞ ifttt.com
  - ☞ zapier.com