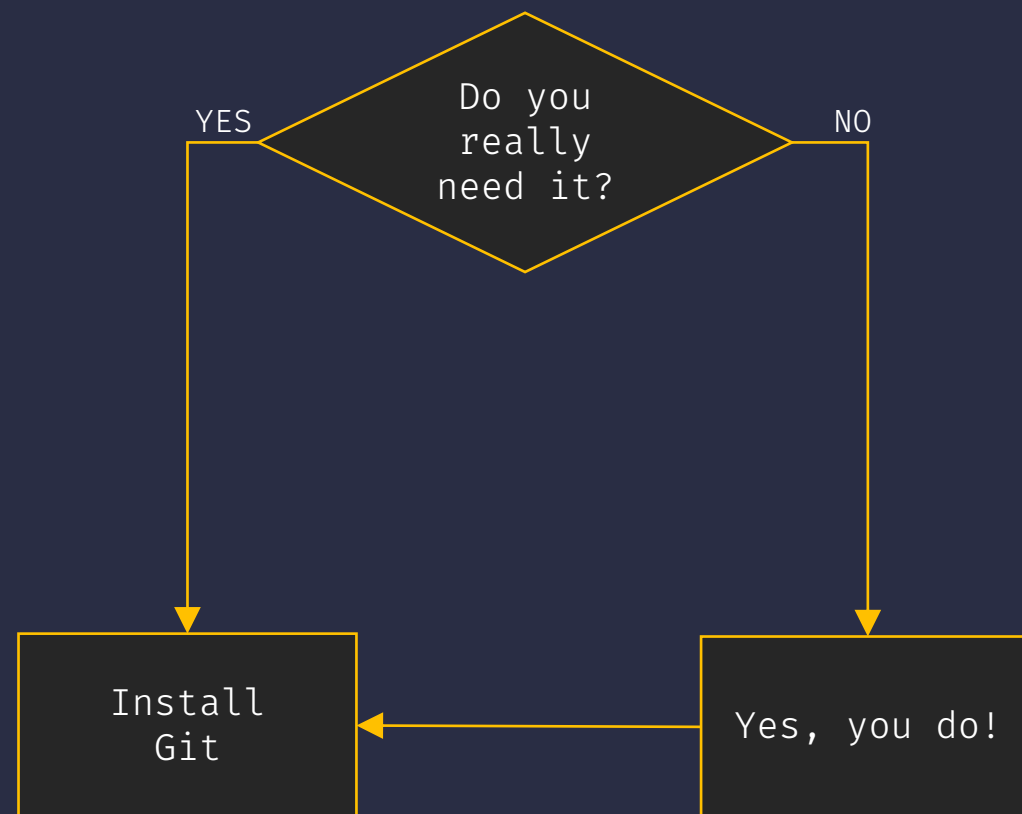
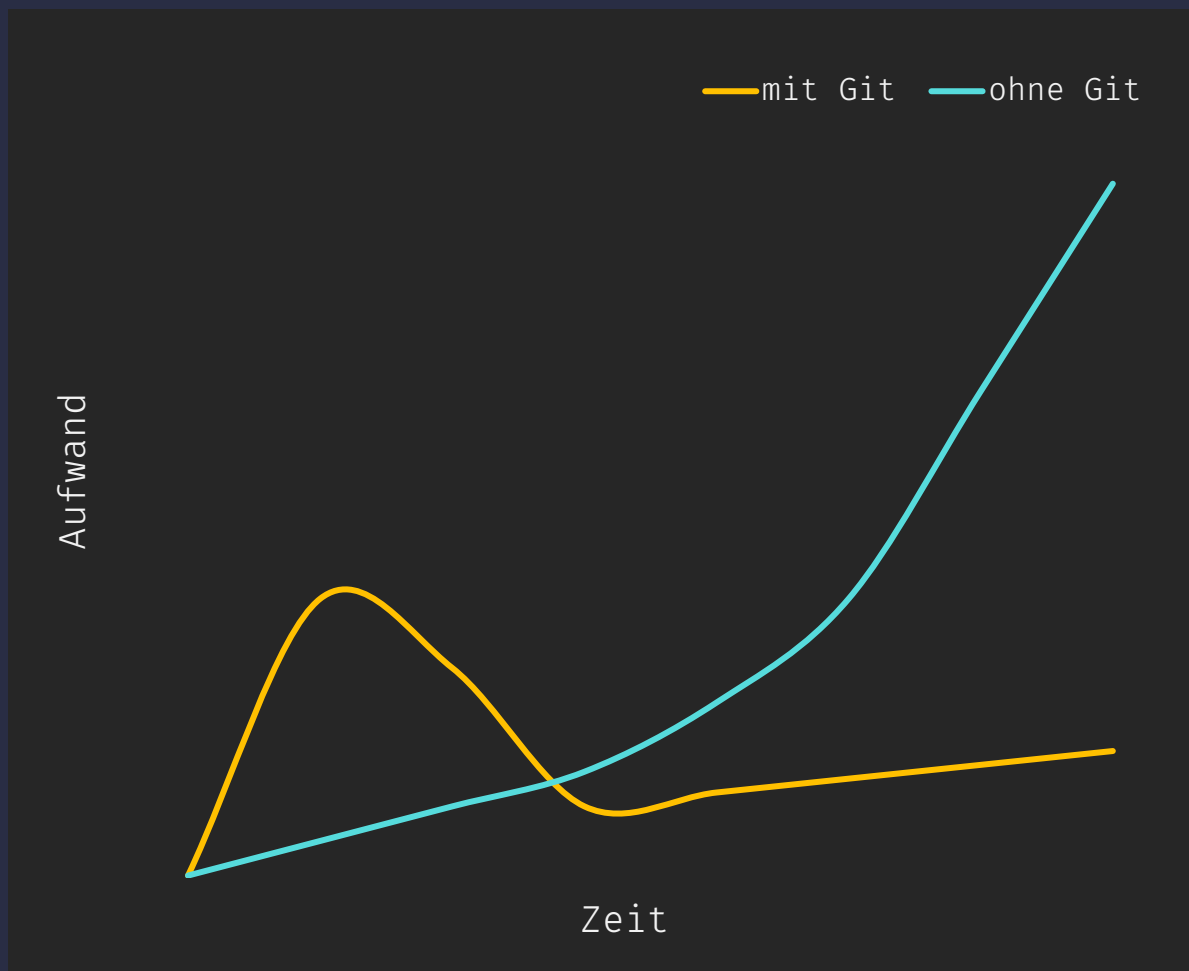




git

{Warum git?;}

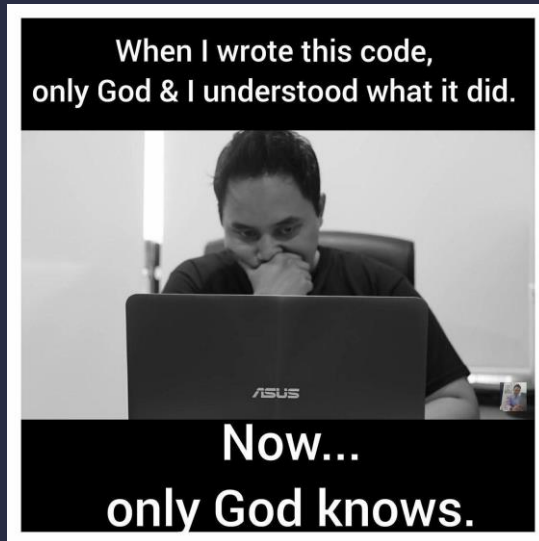
<2/>



{Was ist git;}


- > ursprünglich von **Linus Torvalds** entwickelt
- > **Software** zur Verwaltung und Dokumentation von Veränderungen (**Versionskontrolle**) an Dateien, üblicherweise als System zur **Quellcodeverwaltung** (Source Code Management oder SCM) genutzt
- > **Git != Github**; bekannteste Cloudanbieter: Github (Microsoft), Sourceforge, Bitbucket, GitLab
- > es liegt grundsätzlich Quellcode (Sourcecode) vor, zusätzlich oft Binaries und Dokumentation
- > vereinfacht die **Zusammenarbeit** in Teams und **schnelle Entwicklungszyklen**
- > umfangreiche Integration für **Testautomatisierung** und **Projektmanagement** (Jira, Trello, Jenkins, Wiki)
- > stark in der OSS Kultur verwurzelt: Hervorheben der Autor_innen (contribution), Teilen von Wissen, "Schwarmintelligenz"
- > SCM ist essentieller Bestandteil des Workflows in der Softwareentwicklung (**Basisskill**)
- > auch privat super Quelle und **Spielplatz zum Experimentieren**

Beispiel Dateige- schichte;



Commits on Nov 30, 2020

*bug fixes

 HansenBerlin committed on 30 Nov 2020




5b5b542



Commits on Nov 13, 2020

Finished Expenses Table

 HansenBerlin committed on 13 Nov 2020




5f1bcdb



Commits on Nov 10, 2020

finished decision table

 HansenBerlin committed on 10 Nov 2020




4310bbf



Commits on Nov 9, 2020

Added Decision Planer, Bugfixing Comparison Calculator

 HansenBerlin committed on 9 Nov 2020




9637fcf



Commits on Nov 8, 2020

dashboard changes


 HansenBerlin committed on 8 Nov 2020



4f324fb



Refactoring and debugging of calculations, Added Temporary Data Model...

 HansenBerlin committed on 8 Nov 2020




96ebcc8



Commits on Nov 5, 2020

refactored calculator classes


 HansenBerlin committed on 5 Nov 2020



e76bcf9



Added SVG Icons and production overview to dashboard

 HansenBerlin committed on 5 Nov 2020




43704aa



Commits on Nov 4, 2020

initial

 HansenBerlin committed on 4 Nov 2020



b915fa0



Newer

Older

{ Dateiänderungen ; }

```
Plotly.Blazor.Examples/Models/SetupData.cs

@@ -65,7 +65,7 @@ public SetupData()
65 65      PLTMachinesToReplaceThisRound = FetchTableDataController.ReadValueFromXML("companyProductionData.xml", CurrentGameRound, 1, "PLTMachinesBreakingAfterThis
66 66      ExpenseBoughtMachines = (FetchTableDataController.ReadValueFromXML("companyProductionData.xml", CurrentGameRound-1, 1, "PLTMachinesBoughtThisRound")*1000
67 67      +(FetchTableDataController.ReadValueFromXML("companyProductionData.xml", CurrentGameRound-1, 1, "PCMachinesBoughtThisRound")*3500000);
68 -      ExpenseStorage = Chip1Storage * 0.01 + Chip2Storage * 0.5 + PLTStorage * 10 + PCStorage * 100;
68 +      ExpenseStorage = Chip1Storage * 0.1 + Chip2Storage * 0.5 + PLTStorage * 10 + PCStorage * 100;
69 69      ExpenseRunMachines = PCMachinesAvailableThisRound * 500000 + PLTMachinesAvailableThisRound * 200000;
70 70
71 71      AccountBalance = FetchTableDataController.ReadValueFromXML("marketData.xml", SetupData.CurrentGameRound - 1, 1, "Account");

@@ -94,12 +94,12 @@ public SetupData()
94 94      resetTempData.ResetData();
95 95
96 96      //var listMarketing = new List<double>();
97 -      double marketingMergerd = 0;
97 +      double marketingMerged = 0;
98 98      for (int i = 0; i < 6; i++)
99 99      {
100 -      marketingMergerd += (FetchTableDataController.ReadValueFromXML("marketData.xml", CurrentGameRound-1, i, "Marketing"));
100 +      marketingMerged += (FetchTableDataController.ReadValueFromXML("marketData.xml", CurrentGameRound-1, i, "Marketing"));
101 101      }
102 -      AverageMarketingBudgetAllCompanyys = marketingMergerd / 6;
102 +      AverageMarketingBudgetAllCompanyys = marketingMerged / 6;
103 103      }
104 104      }
105 105      }
```

{ Begriffe (Struktur); }

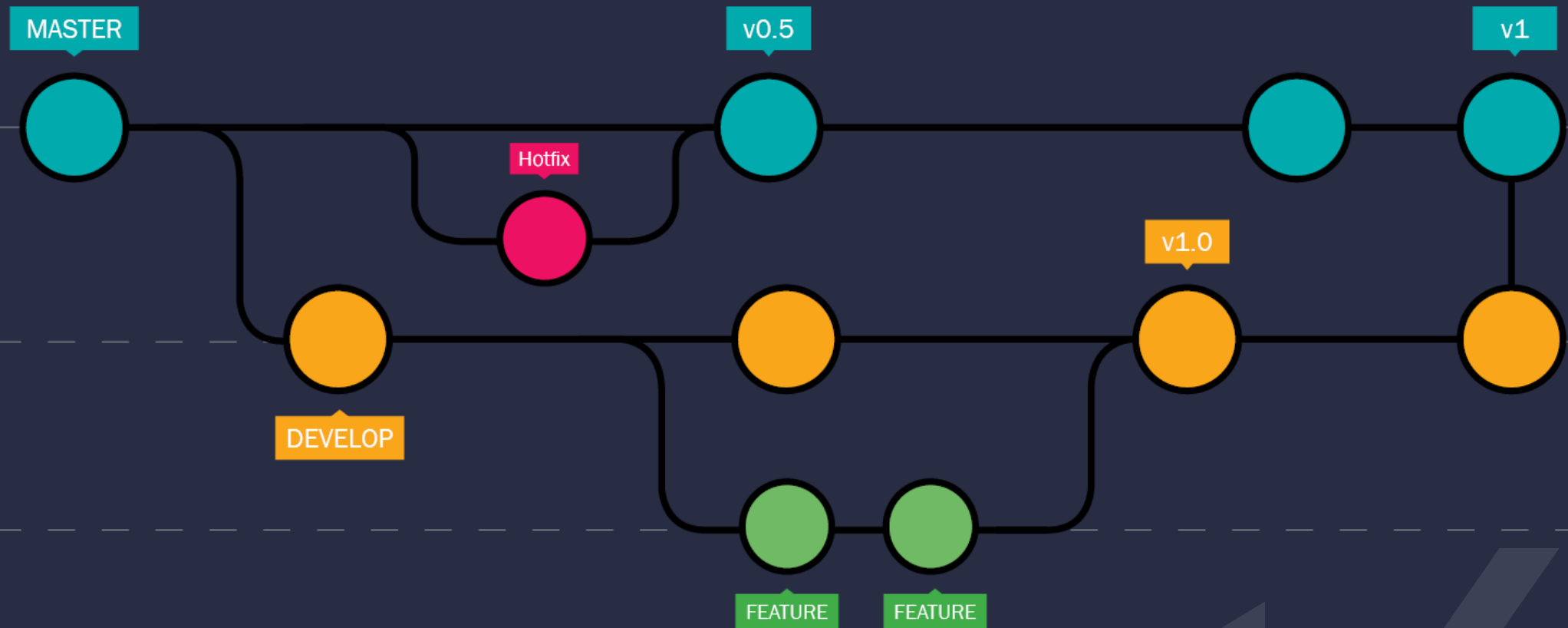
- > repository: Dateicontainer
- > remote: repo auf dem Server
- > local: lokale Kopie des remote
- > origin: ursprüngliches repo
- > master: Hauptverzeichnis („root“)
- > branch: Kinderverzeichnis
- > head: Referenz in einem Branch

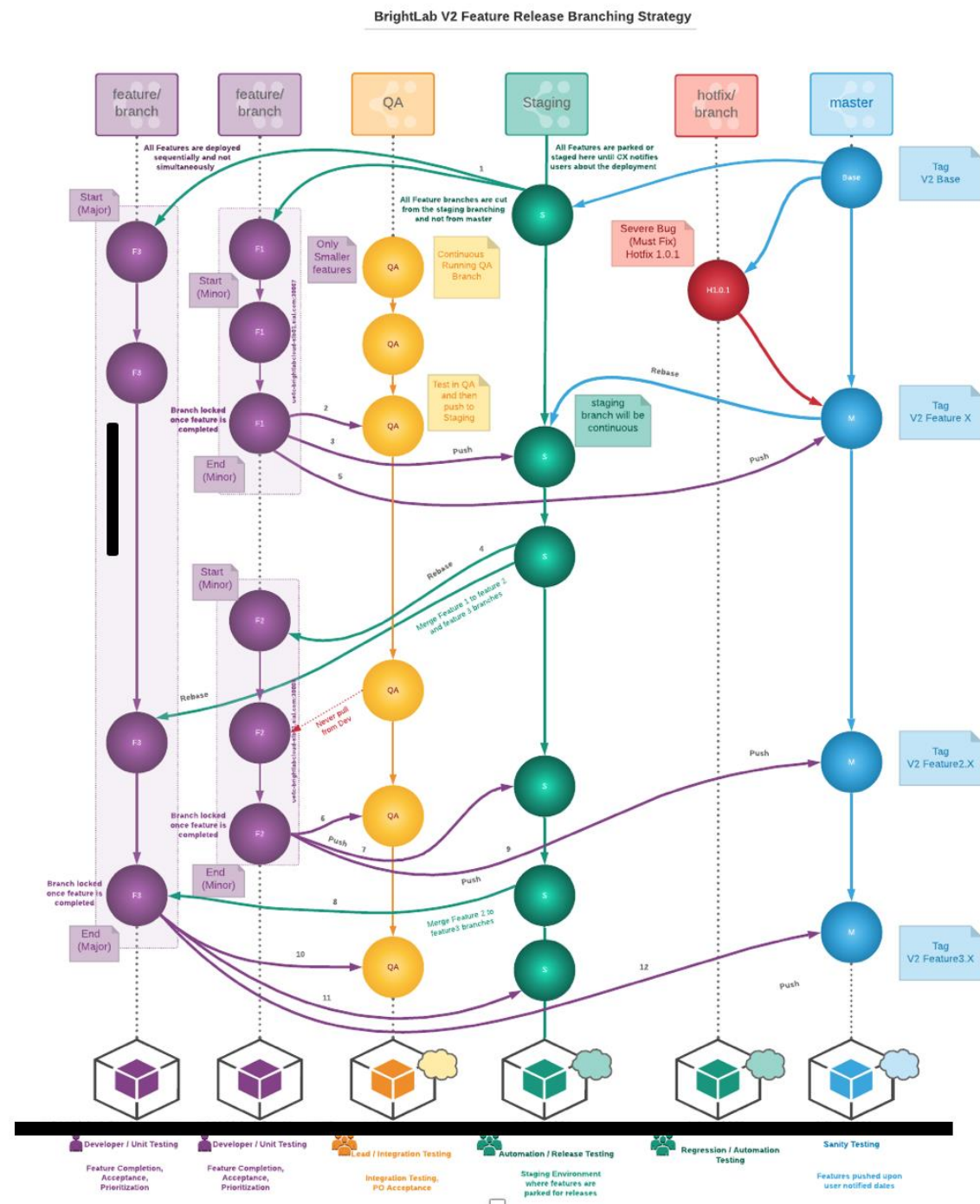


{ Struktur Noob Level ; }



{ Struktur "A good start" - Level ; }





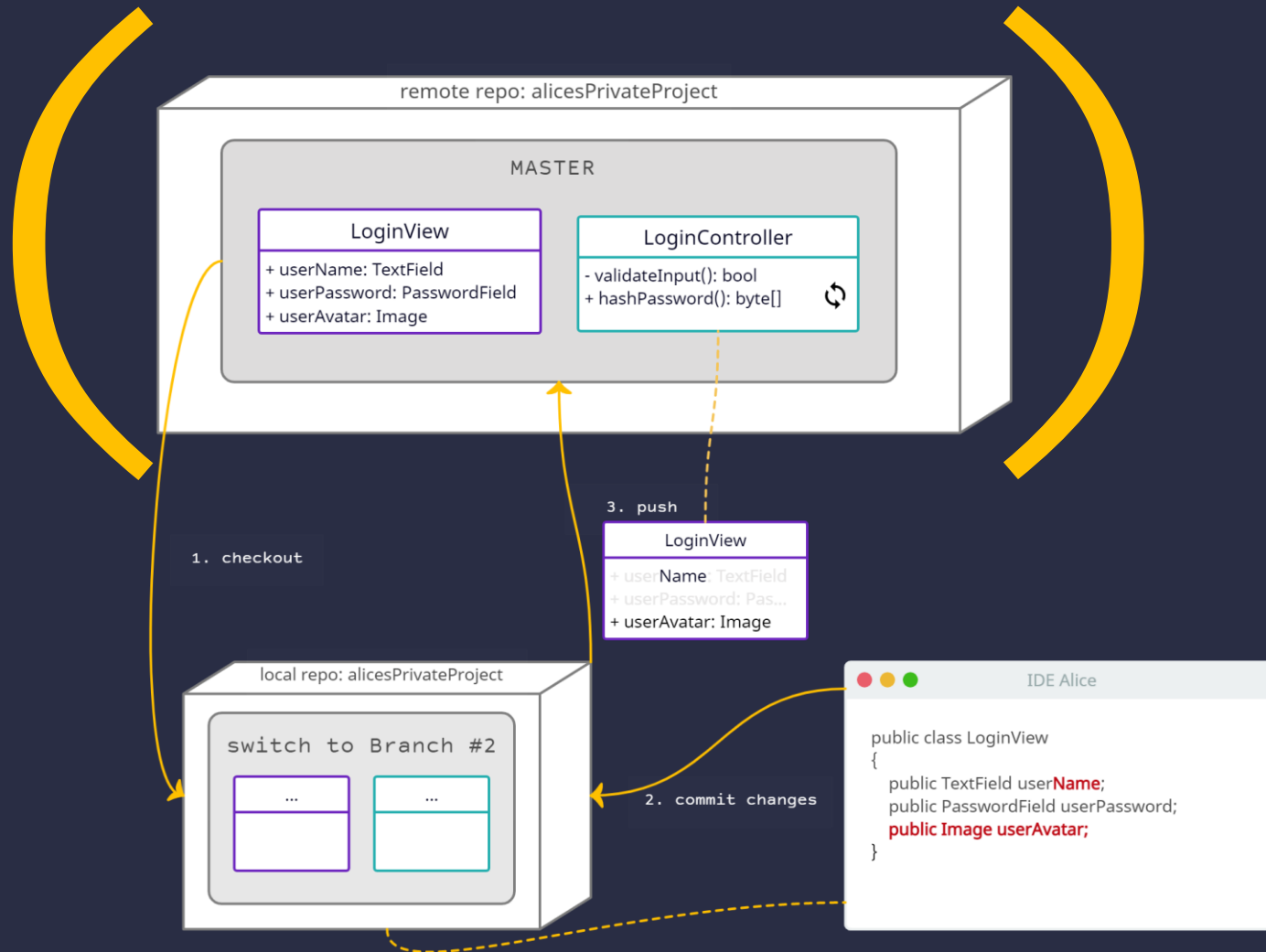
{ Begriffe (Vorgänge); }

- > clone: initiales kopieren von einem remote in einen lokalen workspace (geht auch ohne Account)
- > fetch: aktualisieren eines local repos von remote
- > checkout: Branch wechseln
- > staging: „sammeln“ von geänderten Dateien die commitet werden sollen
- > commit: „(zwischen)speichern“ der Änderungen (nicht Dateien) im local branch
- > push: aktualisieren des remotebranchs (commitmessages und Dateien)
- > pull: aktualisieren eines lokalen branches
- > merge: verschmelzen von zwei Branches
- > checkin: Code „liefern“

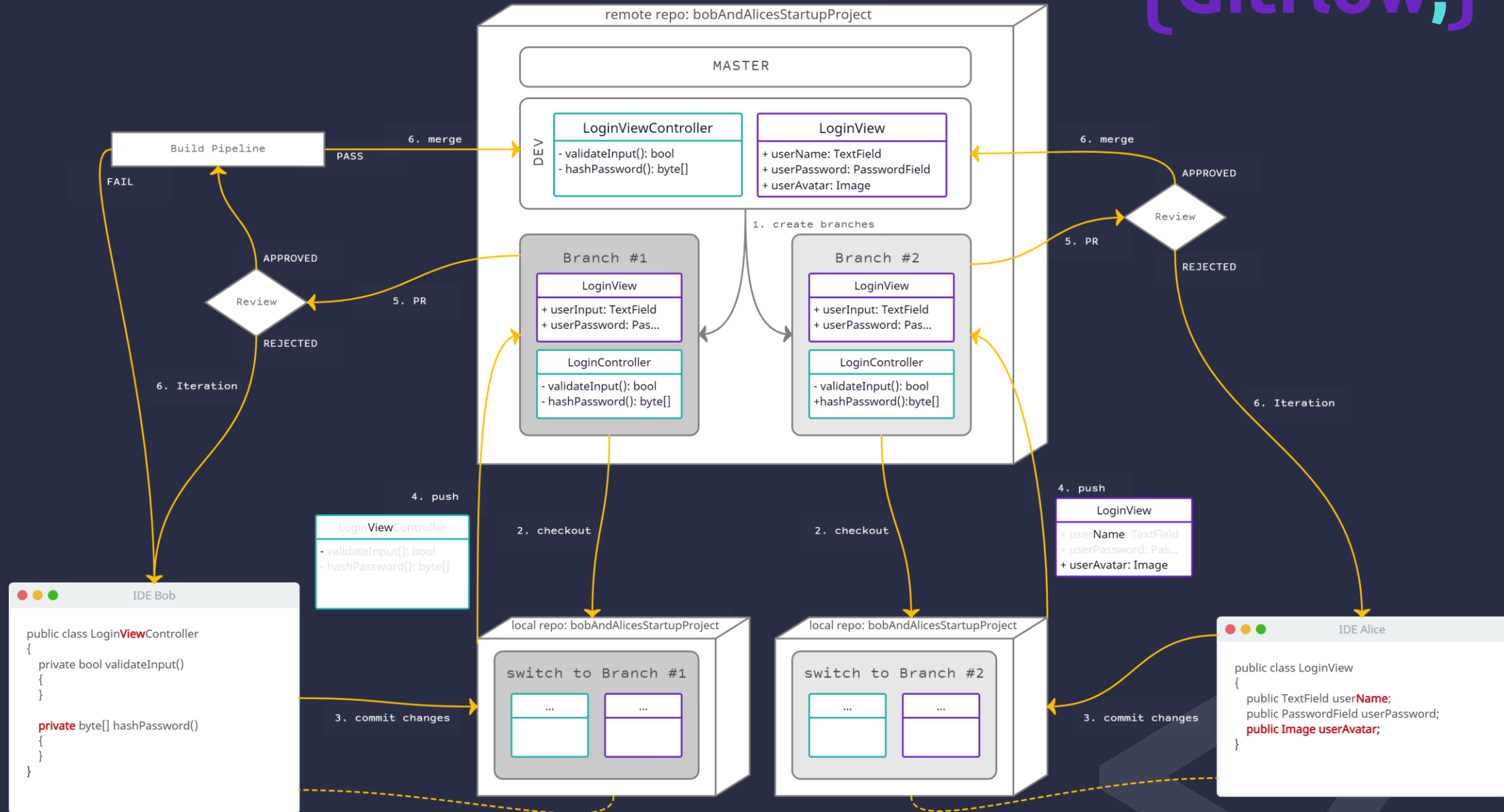
{ Begriffe (sonstige); }

- > **pull request (PR):** “Antrag” den aktuelleren branches in einen zurückliegenden (i.d.R. parent) zu pullen mit dem Ziel zu mergen. Hoffentlich in Verbindung mit einem:
- > **code review:** Feedback zum Code durch andere Entwickler durch Inlinekommentare und Freigabe (approve) oder Zurückweisung (reject oder decline) des PR's.
- > **codefreeze:** keine weiteren merges in den Produktivbranch vor Releases
- > **fork:** Abspaltung eines gitrepos und dadurch Eröffnung eines neuen repos auf Grundlage des alten zur Weiterentwicklung
- > **blame:** Aufschlüsselung der Änderungen nach Autor_in pro Datei
- > **commit message:** kurze Info über Änderungen, folgt oft Vorgaben (vorangestellt: Add, Fix, Refactor... plus message) und/oder beinhaltet die Ticketnummer
- > **build (#):** kompilierte Binärdatei des Sourcecodes, mit jeder (erfolgreichen) Kompilierung (build) wird die Buildnummer inkrementiert
- > **versioning:** != Buildnummer, für Releaseversionen nach unterschiedlichen Schemata (z.B. major.minor.build → 0.12.42)

{einfacher Flow;}



{Gitflow;} <13/>



{Do's;}

- > sinnvolle **Referenzierungen** (PRs, Tickets, Issues, Branches) `#Ticket @Contributor *[]Todo`
- > Erzeugung eines eigenen **Branches pro Issue**/Vorhaben, PR nach Erledigung
- > nicht an tausend Branches gleichzeitig arbeiten
- > **kleinteilig arbeiten**, bestenfalls nur an einer Klasse
- > commits und **push nur in den jeweiligen branch**
- > kurze und **aussagekräftige commitmessages** (ggfs. mit Prefix und/oder Ticketnummer)
- > der eingecheckte Code soll immer besser sein als der ausgecheckte (**Boy Scout Rule**)
- > außer für minifixes (z.B. Typos) PR's **immer mit Review**
- > ggfs. den anschließenden merge an Voraussetzungen knüpfen (erfolgreiche Unittests, Linter, statische Codeanalyse)
- > **klares Feedback in den Reviews**, unabhängig des Erfahrungslevels, aber nicht selber Änderungen vornehmen (you code it, you own it). Unbedingt auch Kleinigkeiten ansprechen, was einmal drin ist wird oft lange nicht mehr angefasst.
- > Mut zur Überarbeitung auch von fremdem Code
- > Lib-Files nur ausnahmsweise, besser Verweise in eigener Datei, z.B. **Maven**
- > Immer einen **stablebranch** garantieren (Master/Main)
- > Einheitliche **Versionierung und Nummerierung** der Builds

{Dont's;}

<15/>

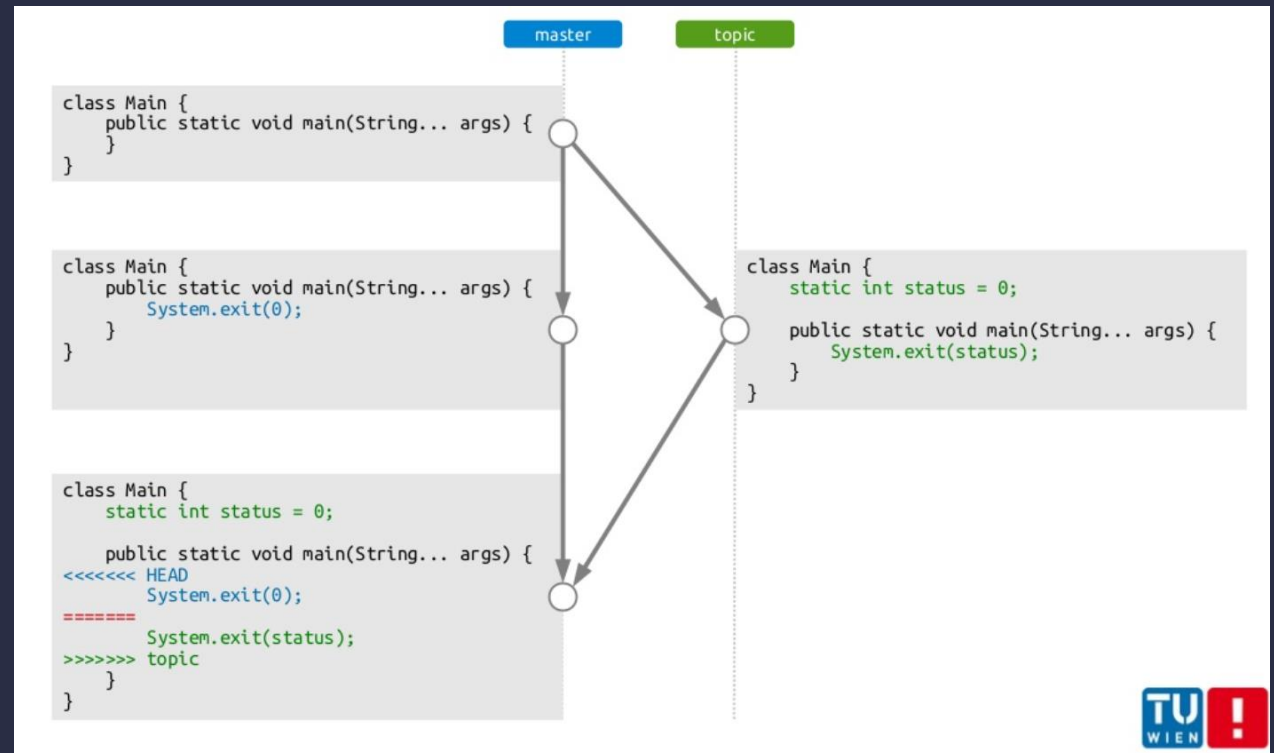
- > Erweiterung der eigentlichen Aufgabe auf andere Klassen (insb. wenn im Team gearbeitet wird)
- > Ändern von „fremden“ Code **außerhalb des Tickets**, weil ihr ihn nicht versteht (Blame!)
- > **Mammut PRs** mit der Erwartung eines sofortigen Reviews
- > Passwörter, Zugangsdaten oder **Keys im Sourcecode** (maximal für POCs in privaten Repos, dann aber später rausnehmen und ändern)
- > **Push in den Master**/direkt auf dem Master arbeiten
- > Direkte **Bearbeitung von Dateien im Browser**, insbesondere nicht in zurückliegenden Branches
- > **push --force** um **Mergekonflikte** zu umgehen



Your branch is behind
'origin/master' by 495 commits,
and can be fast-forwarded.

{ Exkurs: Merge Conflicts; }

- > Beim Zusammenführen von zwei Branches können Konflikte entstehen, **der Merge ist dann nicht möglich**
- > häufigste Ursachen: in zwei Branches vom selben Knoten wird an **denselben Dateien** gearbeitet, der checkin des zweiten Branches führt dann zu Konflikten
- > Bearbeiten des Elternbranches **nachdem in einem davon erstellte Branch Änderungen vorgenommen** und gemergt werden sollen (Abbildung)
- > Lösung: o.g. Ursachen vermeiden, Konflikte in Github lösen oder über Konsolenkommandos (advanced)

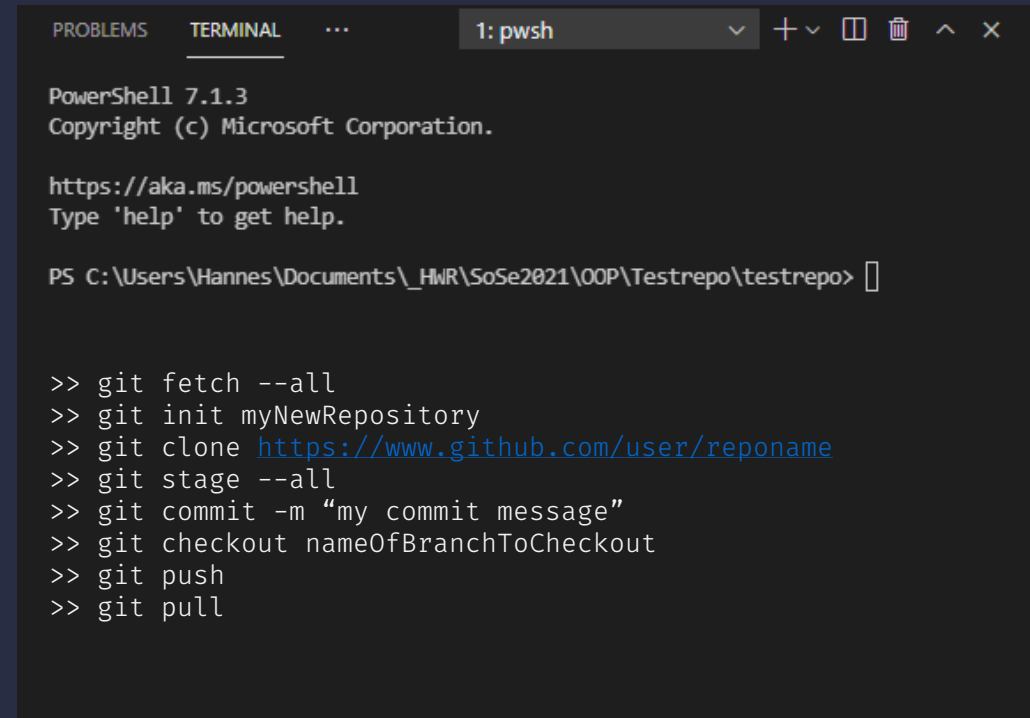


{ Installation/Setup; }

- > **eclipse**: inklusive? wenn nicht: **egit-plugin**
- > **VS Code**: **git installieren**: <https://git-scm.com/downloads>
- > Github Account in der IDE verknüpfen (nicht nötig wenn nur repos geklont werden sollen)
- > Github Account anlegen
- > neues Repo erstellen, URL kopieren (S. 19)
- > Repo in der IDE klonen: (S. 20 & 22)
 - > eclipse: Datei -> import -> remote repository -> clone URI -> smart import -> URL einfügen
 - > VSCode: STRG+Shift+P, git-clone eingeben -> URI(URL) einfügen oder einfach in leerem Projekt auf „clone repo“

{Nutzung in der IDE;}

- > **eclipse**: Rechtsklick auf Projektordner -> **Teams** und/oder eigenes Fenster; Achtung: Wenn in eclipse das SCM Fenster schon auf ist, müssen die Files **vor dem commit gestaged** werden.
- > **VSCode**: Branchicon auf linker Seite -> Dreipunktmenü; automatisches Staging aller Files durch Commit, kann aber auch selektiv über einen Klick auf das „+“ neben der Datei erfolgen
- > Cool kids use **cmd or shell** ;-)
(gängigste Befehle rechts, Vorteil: **geht in allen IDEs und OS**; gängige Herangehensweise zur Installation von Programmen auf Linux)

A screenshot of a PowerShell terminal window. The title bar shows '1: pwsh'. The terminal content includes the PowerShell version (7.1.3), copyright notice, a URL (https://aka.ms/powershell), and a prompt. Below the prompt, a series of git commands are entered and executed: 'git fetch --all', 'git init myNewRepository', 'git clone https://www.github.com/user/reponame', 'git stage --all', 'git commit -m "my commit message"', 'git checkout nameOfBranchToCheckout', 'git push', and 'git pull'.

```
PROBLEMS  TERMINAL  ...  1: pwsh  v  +  -  x

PowerShell 7.1.3
Copyright (c) Microsoft Corporation.


https://aka.ms/powershell
Type 'help' to get help.



PS C:\Users\Hannes\Documents\HwR\SoSe2021\OOP\Testrepo\testrepo>


>> git fetch --all
>> git init myNewRepository
>> git clone https://www.github.com/user/reponame
>> git stage --all
>> git commit -m "my commit message"
>> git checkout nameOfBranchToCheckout
>> git push
>> git pull
```


{Neues Repo anlegen;}

■ Dev ■ WebDev ■ ZSO ■ Stuff ■ Ideas ■ Kali ■ HWR ■ OOP Projekt ■ Hannes' Server ■ Matrizenrechner


 Search or jump to... / Pull requests Issues Marketplace Explore


 + 


Your repository "HansenBerlin/testrepo" was successfully deleted. 


Repositories 


Find a repository...


 HansenBerlin/
altenheimUntilDeath_OOP_Sem2


 HansenBerlin/Kocman-Abschluss

 ant-design-blazor/ant-design-blazor

 HansenBerlin/CalculatorClient

 HansenBerlin/PlanspielWebapp

 HansenBerlin/CalculatorServer


 manuelsidler/blazor-office-addin



Show more


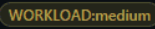
Working with a team?

GitHub is built for collaboration. Set up an organization to improve the way your team works together, and get access to more features.



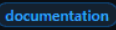
Create an organization

Recent activity 




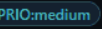
 Zusammenfassung der Systemanalyse 



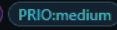
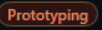
HansenBerlin/altenheimUntilDeath_OOP_Sem2 · You opened this issue 21 hours ago

 Design Idee_I  

HansenBerlin/altenheimUntilDeath_OOP_Sem2 · You commented 21 hours ago

 Erweiterung der Suchlogik auf Minuten   


HansenBerlin/altenheimUntilDeath_OOP_Sem2 · You commented yesterday

 Integration Smarte Suche in Prototyp UI   


HansenBerlin/altenheimUntilDeath_OOP_Sem2 · You were assigned 2 days ago


Show more

All activity


 Stier-09 pushed to HansenBerlin/altenheimUntilDeath_OOP_Sem2 19 hours ago


1 commit to [issue-59-erweiterung-der-suchlogik-auf-minuten](#)

 000e2c5 Fix Bugs


 Stier-09 pushed to HansenBerlin/altenheimUntilDeath_OOP_Sem2 yesterday

1 commit to [issue-59-erweiterung-der-suchlogik-auf-minuten](#)

 dc63b64 Fix Bug (hour of the appointment) and fix the add of possible

 Stier-09 pushed to HansenBerlin/altenheimUntilDeath_OOP_Sem2 2 days ago



2 commits to [issue-59-erweiterung-der-suchlogik-auf-minuten](#)

 65a4514 Refactor

Explore repositories



Tewr/BlazorFileReader

Library for creating read-only file streams from file input elements or drop targets in Blazor.

 C#  294



syncfusion/blazor-samples

Explore and learn Syncfusion Blazor components using large collection of demos, example applications and tutorial samples

 HTML  159

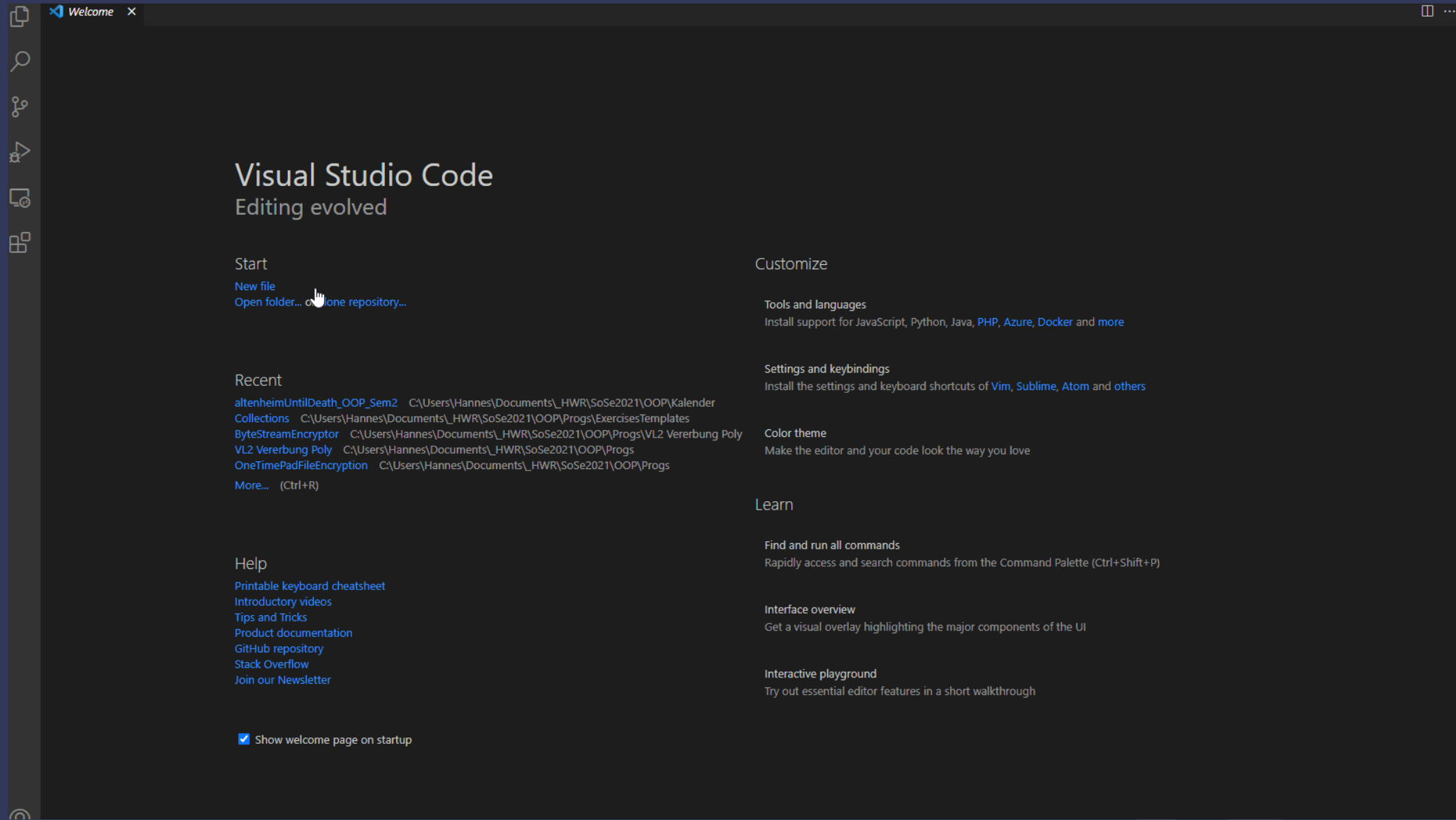
plotly/Plotly.NET

.NET interface for plotly.js written in F#

 F#  136

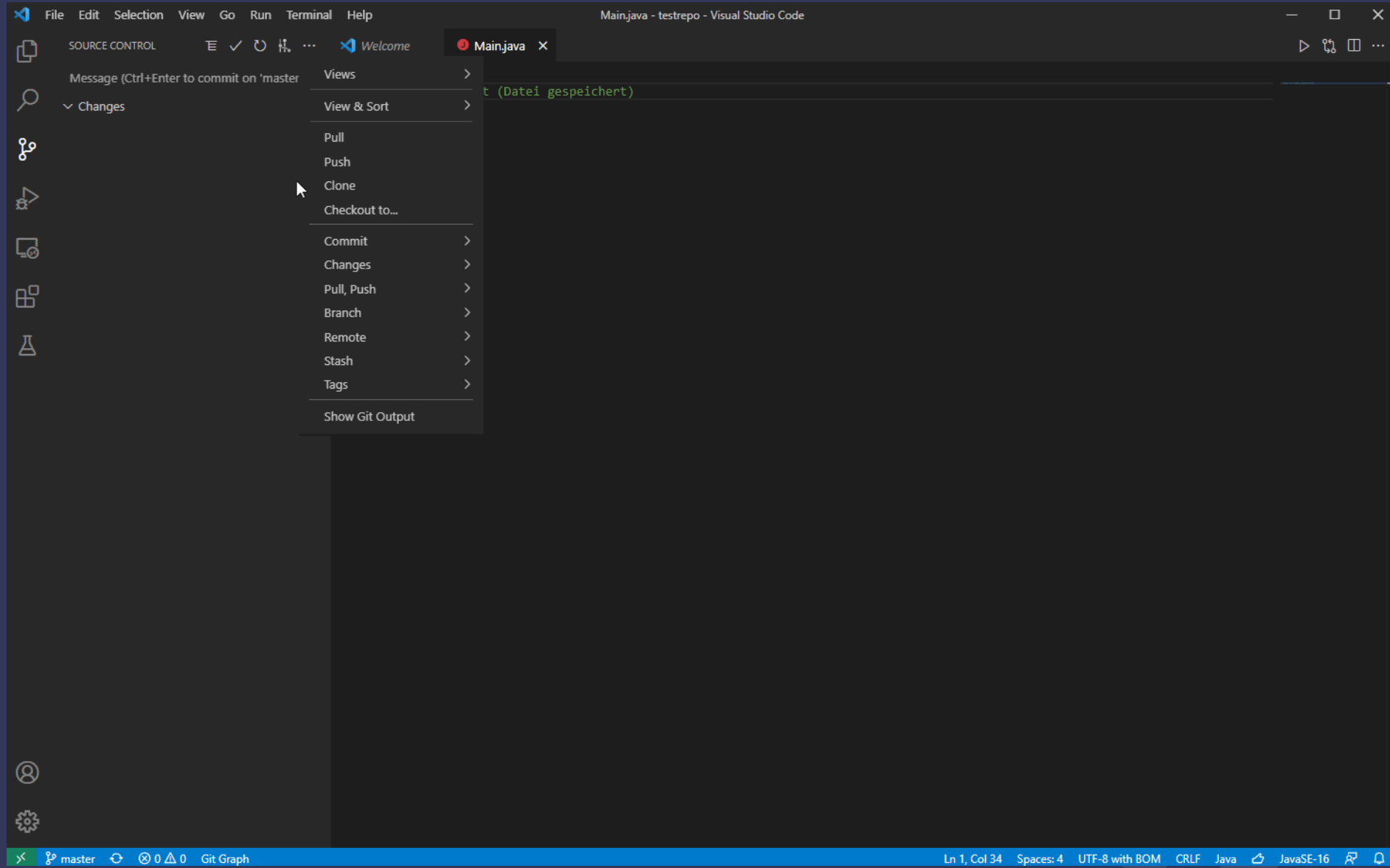
Explore more →

{Repo klonen, commit, push (VS Code);}



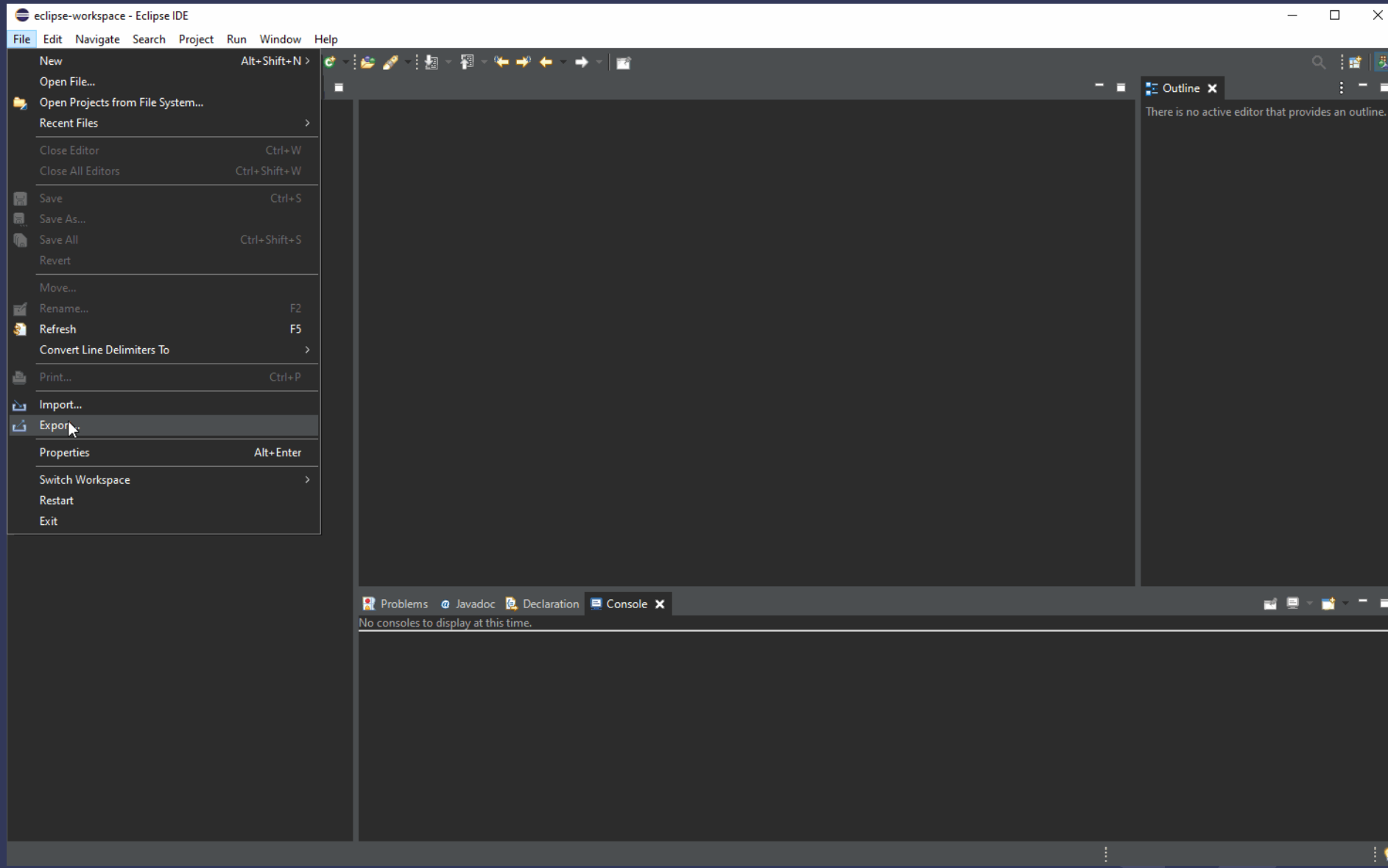
{fetch, commit, push, PR, merge, pull(VS Code);}

<21/>



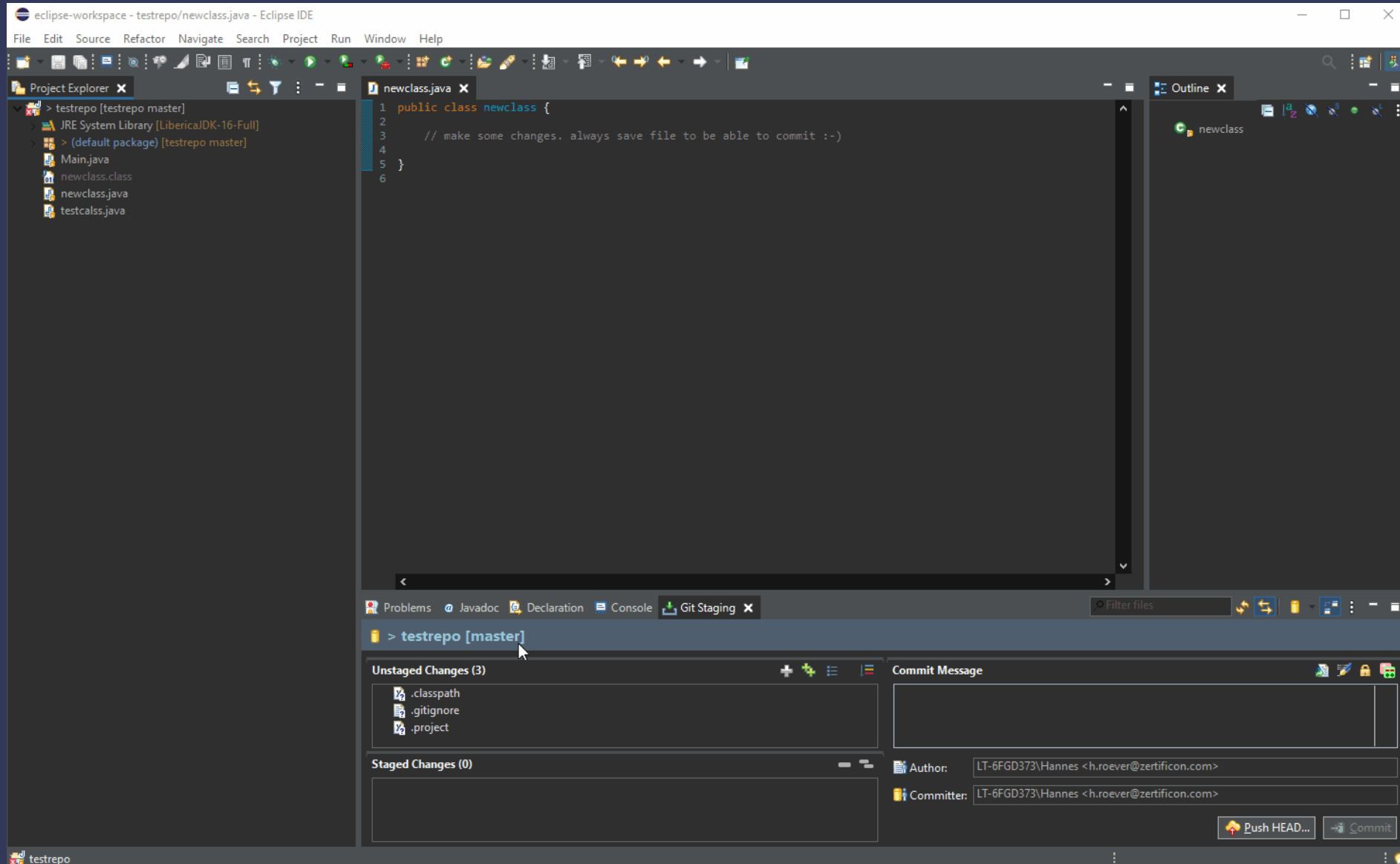
{ Repo klonen, commit, push (eclipse); }

<22/>



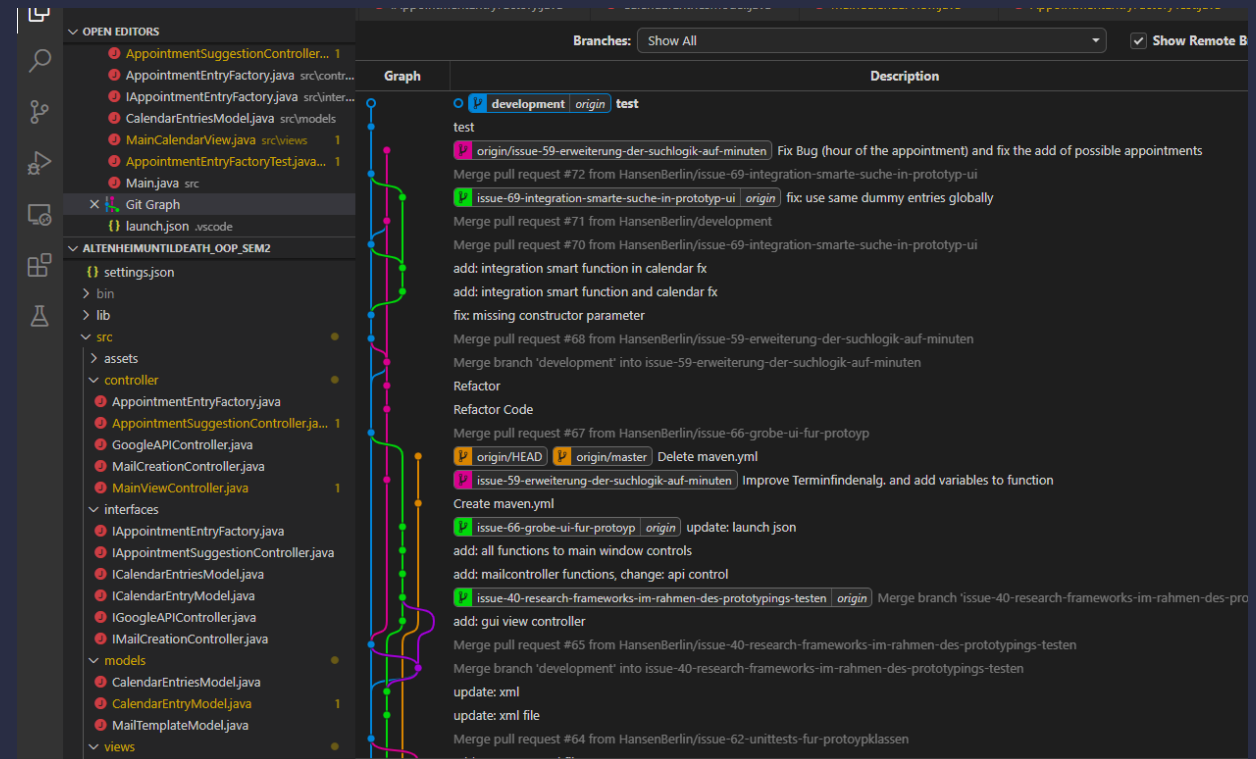
{checkout branch, commit, push (eclipse);}

<23/>



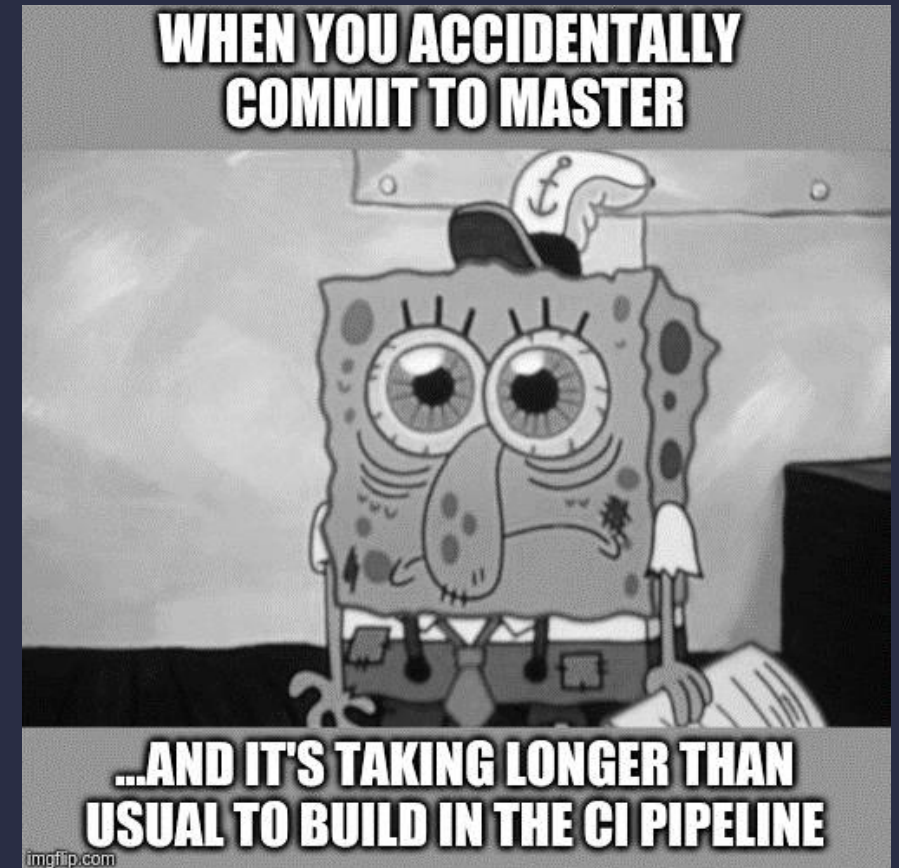
{ Git Clients und Plugins; }

- > Basis: Git
- > alle Befehle gehen über **Konsole/Terminal**, auch die in der IDE
- > über **eingebaute grafische IDE-Funktionalität** (Visual Studio, VS Code, Eclipse)
- > sinnvolle optionale Erweiterung für VS Code: **Git Graph** (Abb.)
- > für erweiterte Funktionen und größere Projekte eignen sich **standalone Clients**: Githb Desktop, Sourcetree, GitKraken

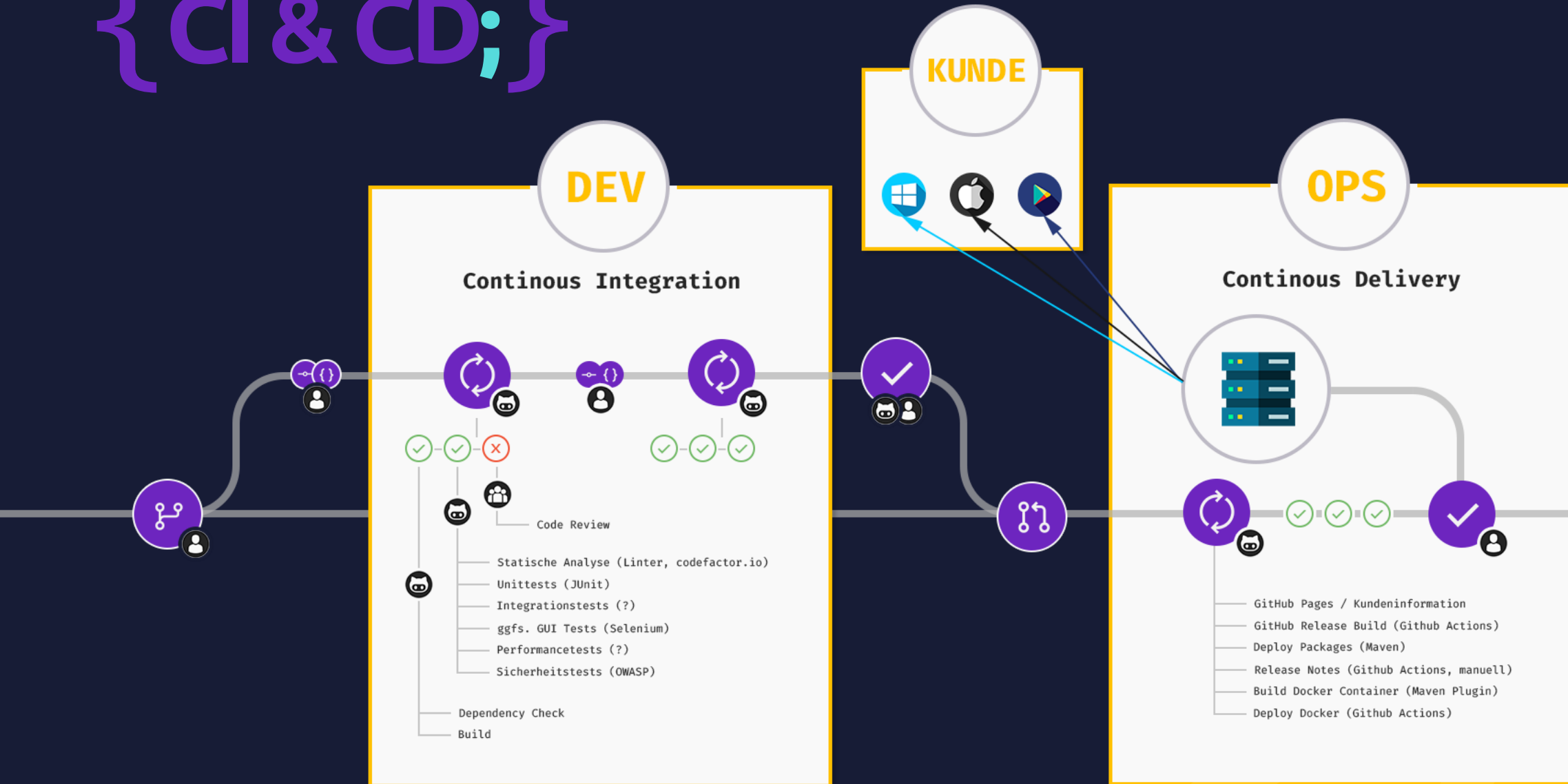


{ Exkurs: CI und CD; }

- > CD (Continuous Delivery) umfasst CI (Continuous Integration)
- > Ziel von CI: kontinuierliche **Integration aller Codebestandteile eines Projekts** (mehrere Entwickler, Sprachen, Frontend, Backend, APIs) durch **automatisierte Builds, Tests und Merges**
- > Ziel von CD: kontinuierliche **Bereitstellung** eines stabilen Branchs und Auslieferung stabiler Builds auf **Deploymentserver**
- > Vorteile: weniger Overhead in Projekten, geringere Fehlerquote, hohe Transparenz gegenüber Kunden, bessere Wartbarkeit
- > **eigentlich geht Agile Entwicklung nicht ohne CI/CD**



{ CI & CD; }



{Tools in/für Github für CI & CD;}

Die folgenden Tools bieten ein gutes Gleichgewicht aus Einsteigerfreundlichkeit und Funktion:

- > Maven oder Gradle (automatische **Paketverwaltung** für Javaprojekte und gute Basis um CI in Github zu nutzen)
- > **JUnit**: ausgereiftes Framework für **Unit Tests** unter Java
- > Mockito: Framework zum Mocken von Objekten für Unittests (fortgeschritten)
- > codefactor.io: Service für **statische Codeanalyse**, einfache Integration in GitHub und für OS Projekte kostenlos
- > **GitHub Actions**: durch Events (push, PR, neues Issue) **getriggerte Skripte** (yaml), die vorkonfigurierte Actions abrufen können und/oder shell-Befehle ausführen
- > Docker & Dockerhub: Deployment der Binarys als **Dockercontainer** (als GitHub Action)
- > GitHub **Releases** (stellt nach eigenem Versionsschema des Sourcecode als Paket bereit)
- > GitHub Pages: automatische Erstellung und Aktualisierung einer **Webseite** zum Repo
- > GitHub Projects: einfache **Projektverwaltung** nach Kanbanstyle

Beispiel GitHub Actions;

✓ Merge pull request #95 from HansenBerlin/development Build and Unit Tests #5 Re-run jobs ...

Summary

Jobs

- ✓ build
- ✓ publish-snapshot

Triggered via push 1 minute ago

HansenBerlin pushed · f558467 master

Status Success Total duration 1m 22s Artifacts -

deploy.yaml
on: push

build 31s → publish-snapshot 26s

<28/>

.factorypath	integrate maven	3 days ago
.gitignore	change gitignore	2 days ago
.project	integrate maven	3 days ago
README.md	Update README.md	9 minutes ago
pom.xml	Merge branch 'master' into development	21 minutes ago

README.md

Build and Unit Tests passing codefactor A version prototype code size 33.8 kB issues 34 open issues 22 closed

Welcome to Altenheim-Kalender

- Coding best practices
- Projektmanagement
- Testing
- Continuous Integration

Current Package

Packages 1

altenheim.calendar.altenheim-kalender
1.0-SNAPSHOT

Contributors 3

- HansenBerlin Hannes
- Stier-09
- dannyneup Danny

Environments 1

github-pages Active

Languages

Java 100.0%