

FreeJDAQ

Visuelle Programmiersprache zur Datenerfassung auf einem Raspberry Pi

David Gawron, Stefan Geretschlaeger, Leon Huck,
Jan Kublbeck, Linus Ruhnke

16. September 2019

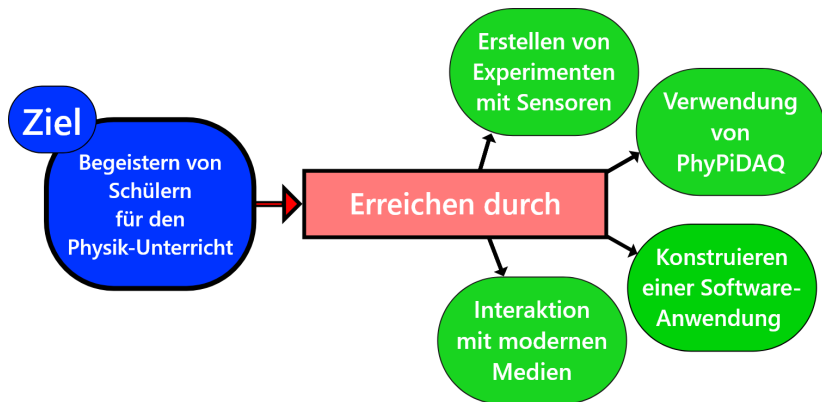
Projektvorstellung

The logo consists of the text "FreeJDAQ" in a bold, black, serif font, centered within a bright yellow rectangular box. The box has a thick black border. The entire logo is set against a light gray background that features a subtle drop shadow effect.

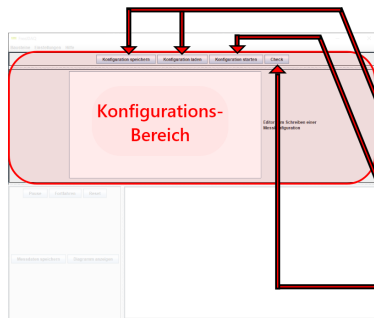
FreeJDAQ

Free Java Data Acquisition

Projektvorstellung



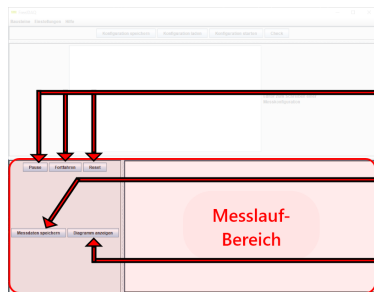
Funktionen des Konfigurations-Bereiches



Funktionen

1. Erstellen und ändern einer Konfiguration
2. Starten von vorhandenen Konfigurationen
3. Überprüfen von Konfigurationen

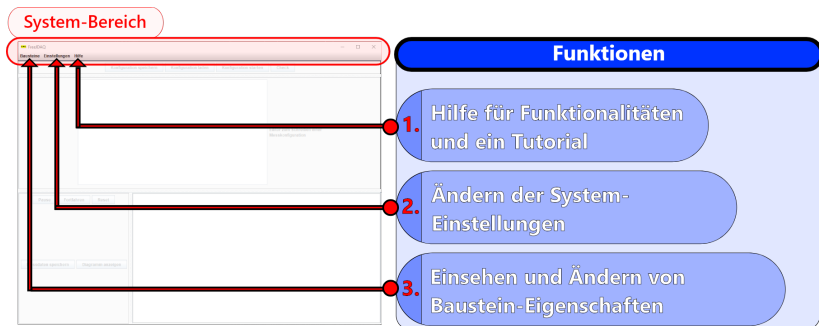
Funktionen des Messlauf-Bereiches



Funktionen

1. Messlauf steuern
2. Messdaten speichern
3. Ergebnisse visualisieren

Funktionen des System-Bereiches

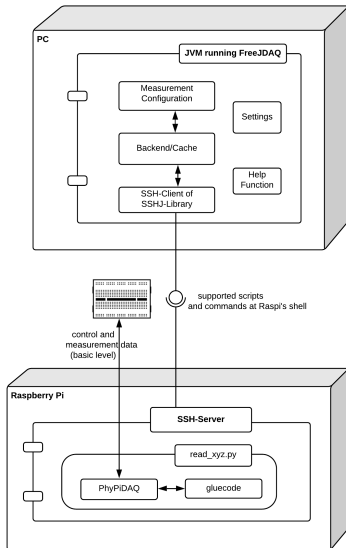


Abgrenzungen

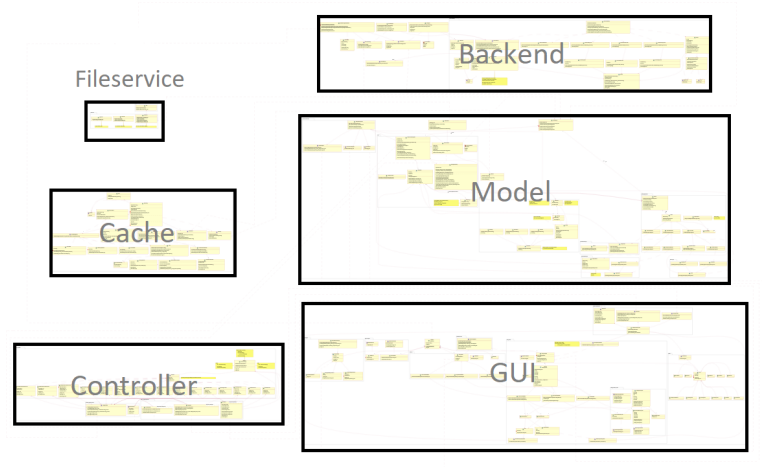
Was unser Produkt nicht enthält:

- Direkte Ansprache der Sensoren. (PhyPiDAQ)
- Visuelle Repraesentation der Messkonfiguration

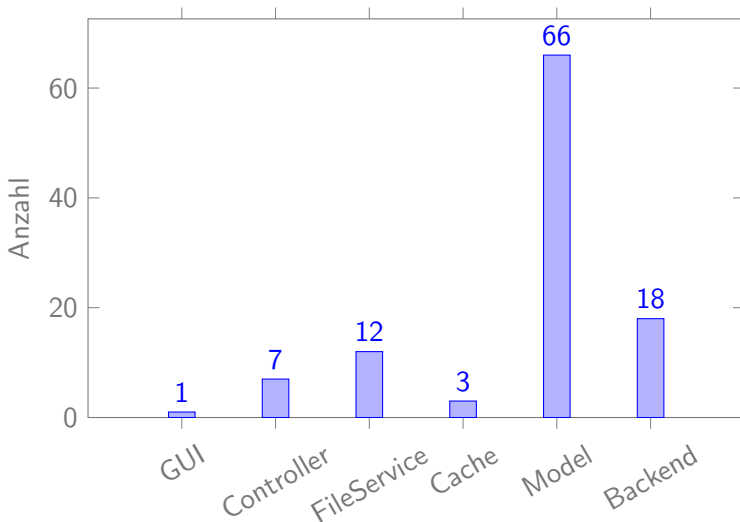
Grundaufbau



Paketdiagramm

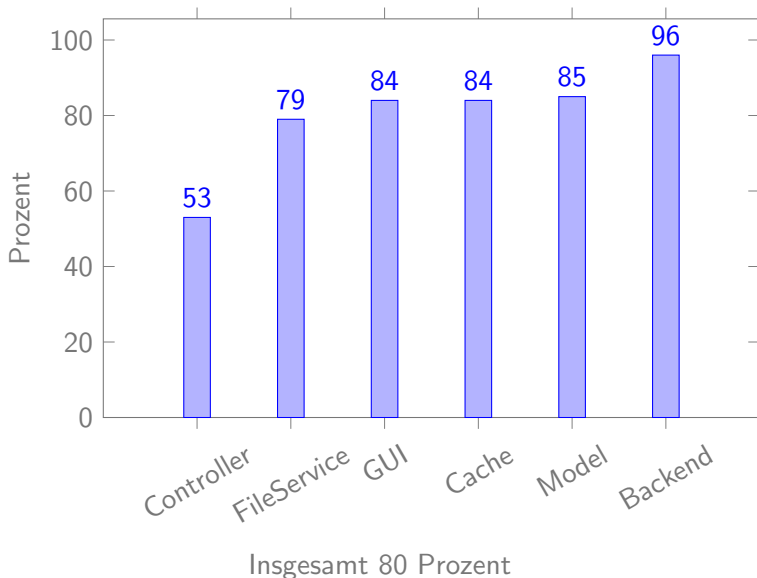


Unit-Tests

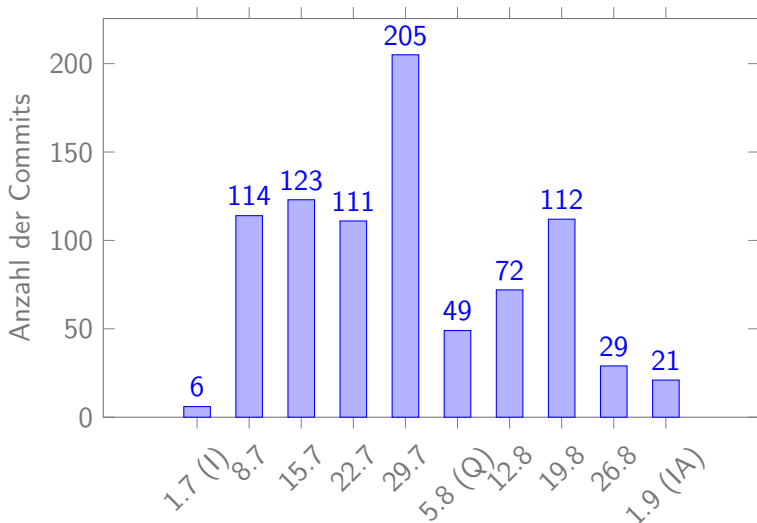


Insgesamt 107 Testcases, zzgl. 33 GUI - Klickstrecken

Testabdeckung



GitHub - FreeJDaq - Commits



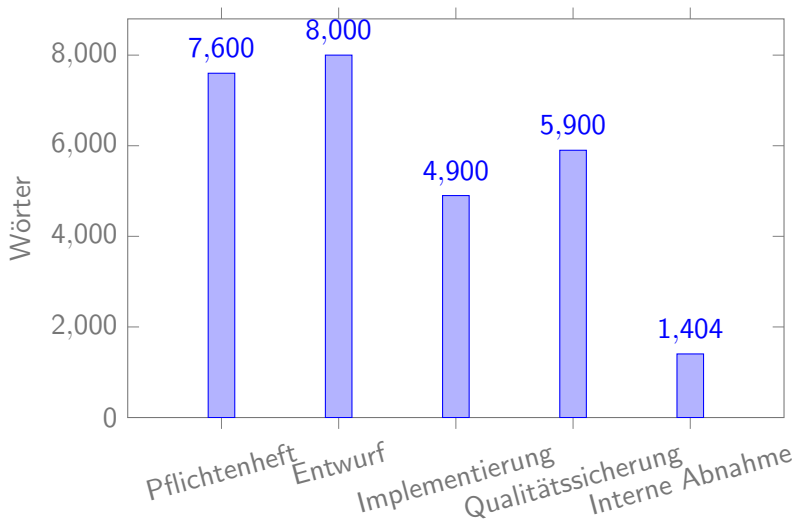
Insgesamt 842 Commits, 54/64 Issues closed, (15.09, 18:00 Uhr)

GitHub - FreeJDaq - Lines of Code

Datei	Anzahl Zeilen
Java	5552
Main	4013
Test	1539
Gesamt (inklusive Kommentar- und Leerzeilen)	12776

Verteilt über 122 Mainklassen und 23 Testklassen

GitHub - DAQDocuments



Insgesamt ca. 27800 Wörter über 994 Commits (15.09, 18:00 Uhr)

Allgemein

UML



Unit-Testing



IDE



Yaml-Editor

SnakeYAML

SSH

SSHJ

Build Management



Statische Codeanalyse



Probleme

- Teamkommunikation in den ersten Phasen
- Nacharbeiten von Fehlern oder Vervollständigung
- Technologiewahl → Technologiewechsel

Was haben wir gelernt

- Phasen planen → Meilensteine, Deadlines setzen und Zuständigkeiten zuteilen.
- Arbeitsverteilung gleichmäßig über den Zeitraum verteilen.
- Meilensteine überprüfen und ggf. Ressourcen verschieben.
- Vor der Implementierung die nötigen Tools aussuchen und in diese einarbeiten.

Livedemo

Quellen

- <https://github.com/osl2/DAQ-Documents>
- <https://github.com/osl2/PhyPiDAQ>
- <https://github.com/GuenterQuast/PhyPiDAQ>
- <http://plantuml.com/de/>
- <https://junit.org/junit5/>
- <https://www.eclipse.org/ide/>
- <https://bitbucket.org/asomov/snakeyaml/src>
- <https://github.com/hieronymus/sshj>
- <https://maven.apache.org/>
- <https://www.eclemma.org/>
- <https://www.eclemma.org/jacoco/>
- <https://www.sonarlint.org/>