

A Collaborative Web IDE for Reasoning about Programs

Qualitätssicherungsdokument

Gruppe 26: Marc Oliver Arnold <marc.arnold@stud.tu-darmstadt.de>
Jonas Johannes Franz Belouadi
<jonasjohannesfranz.belouadi@stud.tu-darmstadt.de>
Anton Wolf Haubner <anton.haubner@outlook.de>
David Heck <david.heck@stud.tu-darmstadt.de>
Martin Kerscher <martin.kerscher@stud.tu-darmstadt.de>

Teamleiter: Tom König <tom.koenig@stud.tu-darmstadt.de>

Auftraggeber: Dr. rer. nat. Richard Bubel <bubel@cs.tu-darmstadt.de>
Dominic Steinhöfel, M.Sc. <steinhoefel@cs.tu-darmstadt.de>
Eduard Kamburjan, M.Sc. <kamburjan@cs.tu-darmstadt.de>
Software Engineering Group
Fachbereich Informatik

Abgabedatum: 10.02.2019

Nachreichung
des Anhangs: 31.03.2019



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Bachelor-Praktikum WS 2018/2019
Fachbereich Informatik

Inhaltsverzeichnis

1. Einleitung	3
1.1. Einführung in KeY	3
1.2. Motivation für KollaborierbaR	3
2. Qualitätsziele	4
2.1. Benutzbarkeit	4
2.1.1. Maßnahme	4
2.1.2. Prozess	4
2.1.2.1. Durchführbarkeit	4
2.1.2.2. Aufgabenstellung	5
2.1.2.3. Fragebogen	6
2.1.2.4. Durchführung	6
2.1.2.5. Auswertung	6
2.2. Wartbarkeit	7
2.2.1. Maßnahme: Code Review	7
2.2.1.1. Prozess	8
2.2.1.2. Auswertung	8
2.2.2. Maßnahme: API Dokumentation	9
2.2.2.1. Prozess	9
A. Anhang	11
A.1. Erste Nutzerstudie	12
A.1.1. Deutsche Aufgabenstellung	13
A.1.2. Englische Aufgabenstellung	15
A.1.3. Deutscher Evaluierungsbogen	17
A.1.4. Englischer Evaluierungsbogen	21
A.1.5. Auswertung	25
A.2. Zweite Nutzerstudie	36
A.2.1. Deutsche Aufgabenstellung	37
A.2.2. Englische Aufgabenstellung	39
A.2.3. Deutscher Evaluierungsbogen	41
A.2.4. Englische Evaluierungsbogen	46
A.2.5. Auswertung	51
A.3. Code Reviews	63
A.3.1. Review Checkliste	64
A.4. Code Anhänge	68
A.5. Tool Konfigurationen	97
A.6. Server HTTP API Spezifikation	117
A.7. Server API Tests	151

A.8. Ausgefüllte Fragebögen	170
A.8.1. Nutzerstudie: Ausgefüllte Evaluationsbögen	171
A.8.1.1. Ausgefüllte Bögen, 1. Studie, deutsch	172
A.8.1.2. Ausgefüllte Bögen, 1. Studie, englisch	293
A.8.1.3. Ausgefüllte Bögen, 2. Studie, deutsch	305
A.8.1.4. Ausgefüllte Bögen, 2. Studie, englisch	434
A.8.2. Code Reviews: Ausgefüllte Checklisten	441
A.8.2.1. Iteration 1	442
A.8.2.2. Iteration 2	447
A.8.2.3. Iteration 3	503
A.8.2.4. Iteration 4	513
A.8.2.5. Iteration 5	536
A.8.2.6. Iteration 6	568
A.8.2.7. Iteration 7	608
A.8.2.8. Iteration 8	634
A.8.2.9. Iteration 9	677

1 Einleitung

KollaborierbaR ist konzipiert als eine Web Applikation, welche es Teams erlaubt, gemeinsam an der formalen Spezifikation und Verifikation von Java Code zu arbeiten. Zu diesem Zweck soll KollaborierbaR auf das bestehende KeY Projekt [1] zurückgreifen. KeY stellt die zur Verifikation notwendigen Funktionen bereit.

KollaborierbaR besteht aus zwei Komponenten: Einer Nutzeroberfläche und einem Server. Die Nutzeroberfläche (im Folgenden Client genannt) bietet einen Editor für Java Code und Spezifikationen mit Syntax Highlighting. Der Editor unterstützt zudem kollaborative Bearbeitung¹ von Quellcode und Verifikation. KeYs assistiertes Führen von Beweisen ist durch Auswahl der betreffenden Spezifikation im Editor möglich. Auch die Durchführung eines automatisierten Beweises wird ermöglicht. Der Beweis wird als Baum visualisiert.

Der Server wird benötigt, um Fehlermeldungen und Warnungen für den editierten Java Code sowie KeYs Verifikationsfunktionen bereitzustellen. Außerdem ermöglicht er die für die Zusammenarbeit notwendige Kommunikation zwischen Clients.

1.1 Einführung in KeY

KeY ermöglicht es Java Entwicklern mittels eines formalen Kalküls automatisch oder assistierend zu beweisen, ob Java Code eine gegebene Spezifikation erfüllt. Eine solche Spezifikation wird mittels einer Erweiterung der Java Modeling Language (JML) innerhalb des zu beweisenen Codes formuliert. KeY extrahiert nötige Informationen aus diesen Quelldateien, bevor es Beweisansätze visualisiert und seine weiteren Funktionen anbietet. Diese umfassen den automatischen oder schrittweise assistierten Beweis und die Visualisierung der Ergebnisse.

1.2 Motivation für KollaborierbaR

Zur Zeit arbeiten vor allem wissenschaftliche Mitarbeiter des Fachgebiets Software Engineering an und mit KeY. Auch Studierende nutzen KeY während der Vorlesung „Formale Methoden im Softwareentwurf“ (FMiSE).

Das gemeinsame Arbeiten verschiedener Personen mithilfe von KeY an Beweisen und Spezifikationen größerer Java Projekte gestaltet sich bisher schwierig. Dateien müssen abwechselnd bearbeitet und manuell geteilt werden. Zusätzlich fallen beim Speichern und Laden größerer Beweisstrukturen längere Wartezeiten an. Auch wird der Arbeitsfluss durch eine Aufteilung auf mehrere Applikationen (Eclipse IDE zum Editieren und KeY Oberfläche für Verifikation) belastet.

Eine Lösung, welche Teamarbeit besser unterstützt und bisherige Funktionen vereint hat daher hohes Potential, die Arbeit mit KeY für wissenschaftliche Mitarbeiter und Studierende zu vereinfachen. Da es sich zudem um eine webbasierte Applikation handelt, wird auch das Teilen von Projekten durch KollaborierbaR automatisiert.

¹ Ein Vorgang, bei welchem mehrere Autoren gemeinsam ein Dokument bearbeiten können, wobei Änderungen jeweils anderer Parteien ohne wahrnehmbare Verzögerungen sichtbar werden.

2 Qualitätsziele

2.1 Benutzbarkeit

KollaborierbaRs Entwicklung ist primär durch Herausforderungen bei der Nutzung von KeY motiviert. Folglich muss KollaborierbaR dem Anspruch eines täglichen Arbeitswerkzeugs für Studierende und wissenschaftliche Mitarbeiter des Fachgebiets genügen. Damit dieses Vorhaben erfolgreich ist, muss die Benutzbarkeit von KollaborierbaR maximiert werden. Im Folgenden wird beschrieben, welche Aspekte von KollaborierbaR wir als entscheidend für die Benutzbarkeit identifiziert haben.

Der bisherige Arbeitsablauf verändert sich maßgeblich, denn statt mehrerer verschiedener Anwendungen wird nur noch KollaborierbaR zum Spezifizieren und Verifizieren von Java Code genutzt. Daher müssen wir sicherstellen, dass Nutzer auch mit der veränderten Arbeitsweise ihre täglichen Aufgaben bewältigen können.

Weiterhin ist zu bedenken, dass KollaborierbaR über zwei verschiedene Zielgruppen verfügt. Die Mitarbeiter des Fachgebiets Software Engineering arbeiten aktiv mit KeY. Sie sind Experten und bringen einen hohen Wissensstand auf dem Gebiet der formalen Verifikation mit. Andererseits werden Studierende, die das Modul FMiSE hören, in dieser Vorlesung zum ersten Mal mit automatischer Programmverifikation und KeY konfrontiert. Sie besitzen Grundwissen über Logik und Formalismen, beispielsweise Kalküle.

KollaborierbaR wird damit vor die Herausforderung gestellt, Nutzer mit verschiedenem Wissensgrad über formale Methoden anzusprechen. Dies lässt sich durch eine leicht erlernbare und intuitive Nutzeroberfläche erreichen.

2.1.1 Maßnahme

Zur Sicherung der Benutzbarkeit von KollaborierbaR werden wir eine Nutzerstudie einsetzen, da sich Benutzbarkeit nicht durch die Entwickler objektiv bemessen lässt. Eine Nutzerstudie gibt direktes Feedback über die Interaktion von Nutzern mit der Applikation und kann somit helfen, übersehene Probleme aufzudecken oder deren Abwesenheit empirisch aufzuzeigen.

2.1.2 Prozess

2.1.2.1 Durchführbarkeit

Die Durchführbarkeit unserer Nutzerstudie hängt von zwei Bedingungen ab: Zum einen müssen wir bis zum Zeitpunkt der ersten Nutzerstudie einen Projektstand erreicht haben, der ein qualitatives Feedback bezüglich unserer Hauptfeatures ermöglicht.

Die zu implementierenden User Stories wurden in Absprache mit dem Auftraggeber von Beginn an derart ausgewählt, dass Grundfunktionen (Projektverwaltung, Editierung, erste KeY Anbindungen) zuerst implementiert werden, statt einzelne Features zu perfektionieren. Auf diese Weise wurde es ermöglicht, bis zum Start der Nutzerstudie einen ausreichenden Projektstand zu erreichen.

Des Weiteren ist es nötig, Probanden aus beiden Nutzergruppen zu befragen. Nur so ist es möglich, einen guten Eindruck zu erhalten, wie flexibel sich KollaborierbaR bei unterschiedlichem Wissensstand einsetzen lässt.

An der Studie nehmen fünf Mitarbeiter des Fachgebiets, darunter auch der Auftraggeber, teil. Weiterhin haben wir in der Vorlesung des FMiSE Moduls und den Übungen für die Nutzerstudie geworben. Bereits 16 Studierende haben sich hierbei für eine Teilnahme bereiterklärt.

2.1.2.2 Aufgabenstellung

Im Rahmen der Nutzerstudie sollen die Probanden einen Aufgabenbogen bearbeiten. Die Aufgabenstellung muss unsere Anwendung auf drei Kernkompetenzen überprüfen:

- Verhalten sich die grundlegenden Funktionen der Entwicklungsumgebung wie Nutzer es von anderen Umgebungen kennen und erwarten? Grundlegende Funktionen sind zum Beispiel das Anlegen und Verändern von Projekten und Dateien.
- Wurde die Funktionalität von KeY in unsere Entwicklungsumgebung nutzerfreundlich integriert? Nutzer der bisherigen KeY Anwendung sollen Funktionen wiedererkennen und so auf ihrer bestehenden Erfahrung aufbauen können. Konkrete Beispiele für diese Integration sind die Folgenden: Bemerken und verstehen die Benutzer das Feedback von KollaborierbaR, z. B. Informationen über den Erfolg oder Misserfolg eines Beweises? Ist die Darstellung und Interaktion mit den Beweisen verständlich für Nutzer, die die Darstellung der bisherigen KeY Oberfläche gewohnt sind?
- Ist das kollaborative Bearbeiten von Dateien intuitiv und verhält es sich so, wie es Nutzer von anderen kollaborativen Editoren kennen?

Daher haben wir drei unabhängige Teilaufgaben konzipiert: Die erste Teilaufgabe soll den Lebenszyklus eines Projektes simulieren. Der Proband wird gebeten, ein Projekt mit mehreren Ordnern anzulegen. Auch soll eine Datei angelegt, geschrieben und gespeichert werden. Hierbei erhält der Proband keine genauen Instruktionen außer vorgegebenem Dateinhalt, sodass auch überprüft wird, wie intuitiv sich die Oberfläche benutzen lässt. Anschließend soll der Proband erst die erzeugte Datei und dann das gesamte Projekt löschen.

Die zweite Teilaufgabe prüft die Interaktion der Probanden mit KeY. Dafür stellen wir ein Projekt mit verschiedenen Java Dateien zur Verfügung. Der Proband soll das Projekt und eine gegebene Datei im Projekt öffnen. In der geöffneten Datei findet der Proband eine Reihe von spezifizierten Methoden. Nun soll der Proband über die integrierten KeY Funktionalitäten versuchen, die Spezifikation automatisiert zu beweisen. Diese Aufgabe testet, ob und wie gut der Nutzer Spezifikationen beweisen kann. Des Weiteren wird getestet, ob der Nutzer sich in einem bestehenden Projekt orientieren kann.

Die dritte Teilaufgabe prüft das kollaborative Editieren von Dateien. Der Proband muss eine gegebene Datei so editieren, dass sie einer Vorlage in der Aufgabenstellung gleicht. Der Betreuer editiert über einen zweiten Computer zeitgleich die Datei. Durch Absprache wird aufgeteilt, wer welchen Teil der Datei editiert. Der Betreuer gibt jedoch keine weiteren Hinweise.

2.1.2.3 Fragebogen

Weiterhin wurde für die Probanden ein Fragebogen entworfen, welcher Fragen zur Benutzbarkeit KollaborierbaRs enthält.

Dieser Fragebogen soll sich für eine formative Evaluation eignen, d.h. wir möchten die Benutzbarkeit konkreter Details der Nutzeroberfläche mittels qualitativem Feedback verbessern [2]. Unter qualitativem Feedback verstehen wir Daten, die es uns erlauben, nicht nur zu erfassen, wie zufrieden der Nutzer mit bestimmten Elementen (z.B. dem Editor) der Nutzeroberfläche ist, sondern auch zu erfassen, genau welche Hürden diesem während der Interaktion aufgefallen sind.

Diese Art des Vorgehens eignet sich besonders für die iterative Verbesserung von KollaborierbaR, sowie für die erneute Überprüfung der Benutzbarkeit. Der Fragebogen kann einfach angepasst werden, um in einer erneuten Durchführung der Studie auf die Effekte von Korrekturen einzugehen, siehe Auswertung, Sektion 2.1.2.5.

Die Zufriedenheit mit den einzelnen Elementen der UI messen wir durch Multiple-Choice Fragen, deren Antwortmöglichkeiten nach unterschiedlicher Zufriedenheit abgestuft sind. Hürden werden identifiziert durch die Freitextfelder im Fragebogen zu den einzelnen Aufgabenteilen.

Um einer Verfälschung der Daten vorzubeugen, wird auf die Vermeidung häufiger Fehler wie sog. „Leading Questions“ achtgegeben. Bei „Leading Questions“ handelt es sich um Fragestellungen an den Probanden, welche Erwartungen der Autoren implizieren [3].

Um zu sichern, dass der Fragebogen all diese Kriterien erfüllt, wird er vor der Durchführung der Studie vom gesamten Entwicklerteam auf deren Einhaltung kontrolliert.

2.1.2.4 Durchführung

Die Nutzerstudie erfolgt innerhalb von zwei Zeiträumen im Jahr 2019 und wird durch das Entwicklerteam durchgeführt. Die Zeiträume sind Anfang Februar und Anfang März eingeplant.

Die Studie soll anhand einer Reihe praktischer Aufgaben für den Probanden in ca. 25 Minuten durchgeführt werden.

Hierbei gibt ein Betreuer aus dem Entwicklerteam die Aufgabenstellungen (vgl. Sektion 2.1.2.2) aus. Nach Bearbeitung der Aufgaben füllt der Proband den Fragebogen zur Evaluation der Benutzbarkeit KollaborierbaRs aus.

Während der Bearbeitung beobachtet der Betreuer zudem die Interaktionen des Nutzers mit KollaborierbaR und notiert Auffälligkeiten informell. Dabei kann es sich z.B. um Kommentare zur Nutzerfreundlichkeit handeln, die der Proband mündlich äußert. Dies verhindert, dass beim Ausfüllen des Fragebogens Details vergessen werden.

2.1.2.5 Auswertung

Auch die Auswertung der Nutzerstudie erfolgt durch das Entwicklerteam und beginnt direkt nach Ende der Durchführung. Mittels der ausgefüllten Fragebögen und den informellen Notizen des Betreuers möchten wir die folgenden Aspekte evaluieren:

Vermissten die Nutzer eine Funktion? Wurden bestimmte Funktionen nicht durch die Nutzer verwendet, z.B. sind Funktionen zur Anpassung der Oberfläche zu versteckt? Weist die Interaktion mit bestimmten UI Elementen, beispielsweise der Sidebar, besonders schlechte Ergebnisse

in den Multiple-Choice Fragen auf oder wurde negativ kommentiert? Sind Routineaufgaben wie das Starten/Fortsetzen von Beweisen zu umständlich realisiert, beispielsweise aufgrund ungeeigneter Menüführung?

Um Aufkommen dieser Aspekte zu identifizieren, wird sämtliches Feedback durch das Entwicklerteam gemeinsam gesichtet. Durch Abstimmung wird entschieden, ob und auf welche Weise ein Problem korrigiert werden soll. Die Korrektur wird hierbei als eine User Story formuliert.

Die neu erstellten User Stories werden in Absprache mit dem Auftraggeber dann spätestens bis zur zweiten Iteration der Studie implementiert. Durch eine zweite Iteration wird daraufhin überprüft, ob die zuvor identifizierten Aspekte verbessert wurden. Um sicherzustellen, dass die als problematisch identifizierten Aspekte verbessert wurden, gibt es in der zweiten Iteration der Nutzerstudie explizite Fragen zu diesen Aspekten. Zusätzlich wird geprüft, ob durch neue Funktionalitäten keine neuen Problematiken entstanden sind. Andernfalls werden entdeckte Mängel erneut korrigiert.

Idealerweise werden diese Iterationen so lange wiederholt, bis keine Probleme bei der Nutzung KollaboriebaRs mehr existieren. Bei realistischer Betrachtung der verfügbaren Zeit ist nun aber keine weitere Nutzerstudie zur erneuten Überprüfung der Korrekturen mehr möglich. Allerdings sind bei Weiterführung des Projekts dem Auftraggeber die Ergebnisse der Studien und identifizierte Schwachpunkte bekannt. Mit den nötigen Werkzeugen ausgestattet (Fragebögen und Erläuterung des Verfahrens in diesem Dokument), kann er bei Bedarf ein neues Entwicklerteam mit weiteren Iterationen der Studie beauftragen.

2.2 Wartbarkeit

KollaborierbaR soll Basis einer Entwicklungsumgebung werden, welche in Zukunft die bisherige KeY Anwendung ablöst. Der Funktionsumfang, der für diese Ablösung notwendig ist, ist jedoch im Rahmen des Bachelorpraktikums nicht erreichbar. KollaborierbaR soll nach Abschluss des Bachelorpraktikums lediglich Grundfunktionalitäten von KeY bereitstellen können. Die Möglichkeit, KollaborierbaR problemlos weiterentwickeln zu können, ist deshalb essentiell für den langfristigen Einsatz. Daher müssen wir sichern, dass die Einarbeitung neuer Entwicklerteams und die Fortführung des Projekts gut möglich sind.

Es muss neuen Entwicklern möglich sein, unseren Code zu verstehen, anzupassen und die Effekte von Anpassungen auf den Rest des Codes zu verstehen. Andernfalls könnten hohe Kosten für eine langwierige Einarbeitung und Weiterentwicklung entstehen. Im schlechtesten Fall ist eine Einarbeitung nicht möglich und die Anwendung müsste neu entwickelt werden.

2.2.1 Maßnahme: Code Review

Die Wartbarkeit einer Software wird durch viele verschiedene Faktoren beeinflusst. Darunter befinden sich unter anderem die Dokumentation bzw. Kommentierung der Implementation, lesbarer und verständlicher Code sowie die Modularisierung in klar getrennte Verantwortlichkeiten.

Um den Aufwand im Rahmen des Bachelorpraktikums begrenzen zu können, nutzen wir daher Code Reviews. Diese erlauben es uns, einen realistischen Anteil der Umsetzung dieser Faktoren festzulegen und zu überprüfen. Zudem verringern wir den Aufwand für die Durchführung, indem wir Anteile des Reviews soweit möglich automatisieren (z.B. Einhaltung von Codingstandards).

2.2.1.1 Prozess

Unser Einsatz von Code Reviews läuft folgendermaßen ab: Unter Einsatz der Git Versionsverwaltungssoftware wird ein Master-Branch als stabile Version der Applikation gepflegt. Jede neue Funktionalität muss von dieser getrennt entwickelt werden. Bevor sie in die stabile Version integriert werden darf, wird ein Code Review durchgeführt. Das Review wird hierbei durch ein anderes Teammitglied (im Folgenden „Reviewer“ genannt) durchgeführt. Der Reviewer sollte nicht an der Entwicklung der Funktionalität beteiligt gewesen sein. Dies ist in der Subjektivität von Faktoren wie der Lesbarkeit des Codes begründet. Wenn die Kommentierung und Dokumentation beim Prüfer zum Verständnis des Codes führt, so dient dies bereits als wichtiger Indikator für dessen Verständlichkeit.

Um einen gemeinsamen Qualitätsstandard über alle Reviewer hinweg zu schaffen, nutzen wir eine Checkliste. Diese ist in mehrere Sektionen unterteilt und geht auf folgende Aspekte ein:

Zunächst überprüft der Reviewer vor der Inspektion, ob die eingereichte Funktionalität sämtliche Überprüfungen einer automatisierten Build-Pipeline besteht. Dabei handelt es sich um eine Prüfung auf Formatierung (mittels CheckStyle und Prettier [4, 5]), statische Analyse des Codes (mittels tslint, eslint, PMD und SpotBugs [6–9]), dessen Kompilierung und Tests der Server API (siehe Sektion 2.2.2). Ergibt sich in nur einem dieser Schritte ein Fehler, so gilt das Review als nicht bestanden. Dieser Abschnitt des Reviews ist bezüglich der Wartbarkeit relevant, da die automatische Überprüfung zur Einhaltung von Codingstandards (Google Java Code Guidelines für den Server, Prettier Formatter Regeln für den Client [5, 10]) der besseren Lesbarkeit dient und die statische Analyse Fehlern vorbeugt.

Nach einem Test der neuen Funktionalität und einem Vergleich, ob diese den Akzeptanzkriterien der User Stories entspricht, folgt nun die eigentliche Inspektion. Die Checkliste stellt hier Anforderungen an die Kommentierung des Codes sowie die Dokumentation der Serverschnittstellen (siehe Sektion 2.2.2).

Auch werden Fragen bezüglich Prinzipien objektorientierten Designs gestellt. Um den Arbeitsaufwand des Reviews in einem realistischen Rahmen zu belassen, wird hier nur auf ausgewählte Kriterien eingegangen. Beispielsweise ist der Code auf die Einhaltung des Single-Responsibility-Prinzips zu überprüfen. Die Auflistung aller Anforderungen der Checkliste befindet sich im Anhang. Diese Kriterien zielen auf die Kommentierung und Modularisierung des Codes ab, sodass dessen Verständlichkeit erhöht wird.

Auch hier gilt das Review als nicht bestanden, sobald nur eine Anforderung der Checkliste verletzt wird. Um Korrekturen zu beschleunigen, muss der Reviewer den Grund für die Verletzung eines Kriteriums kommentieren.

2.2.1.2 Auswertung

Besteht das geprüfte Arbeitsergebnis das Review nicht, so hat der Entwickler die durch den Reviewer angemerkten Mängel zu korrigieren und muss die korrigierte Implementation erneut für ein Review einreichen. Dieser Zyklus ist so oft zu wiederholen, bis das Review bestanden wurde, woraufhin die Funktionalität in die stabile Version der Applikation integriert wird.

2.2.2 Maßnahme: API Dokumentation

Als Web Applikation kommuniziert KollaborierbaRs Client über eine HTTP API mit dem Server. Für die meisten Funktionalitäten KollaborierbaRs ist es notwendig, auf diese Schnittstelle zuzugreifen oder sie zu erweitern. Auch in Zukunft ist für Anpassungen von KollaborierbaR gewiss, dass die Entwickler diese Schnittstelle nutzen und sich intensiv über diese informieren müssen.

Eine standardisierte, zentrale Dokumentation erleichtert also die zukünftige Wartung sowohl des Servers, als auch des Clients von KollaborierbaR. Wir haben uns daher für die Pflege einer OpenAPI 2.0 Beschreibung¹ der Server API entschieden.

Diese wird über die genannten Vorteile hinaus zudem in einem maschinenlesbaren Format (YAML) formuliert. Somit können wir sie zur automatischen Generierung einer übersichtlichen webbasierten Dokumentation für Entwickler verwenden.²

2.2.2.1 Prozess

Für jedes Feature, welches eine Änderung oder Erweiterung der Funktionalitäten des Servers erfordert, aktualisiert der zuständige Entwickler die OpenAPI 2.0 Beschreibung. Die Änderungen der Beschreibung erfolgen, bevor der Entwickler seine neue Funktionalität zum Review einreicht.

Die Einhaltung der Spezifikation durch den Server wird mittels automatisierter Tests der Schnittstelle sichergestellt, welche ebenfalls durch den Entwickler bereitgestellt werden. Weiterhin wird die OpenAPI 2.0 Beschreibung, die Vollständigkeit der Tests und die korrekte Nutzung der API durch den Client als Teil der Code Reviews überprüft.

¹ <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

² Für weitere Tools siehe <https://openapi.tools/>

Literaturverzeichnis

- [1] Offizielle Seite des KeY Projekts. <https://www.key-project.org/>. Zugegriffen: 30.11.2018.
- [2] Jakob Nielsen. Usability engineering. chapter 6, page 170. Academic Press, 1993.
- [3] Artikel über Leading Questions. <https://www.nngroup.com/articles/leading-questions/>. Zugegriffen: 30.11.2018.
- [4] Offizielle Seite des CheckStyle Werkzeugs für die Prüfung der Einhaltung von Coding Standards. <http://checkstyle.sourceforge.net/>. Zugegriffen: 10.2.2019.
- [5] Offizielle Seite des Prettier Code Formatierers. <https://prettier.io>. Zugegriffen: 8.2.2019.
- [6] Offizielle Seite des tslint Linters. <https://palantir.github.io/tslint/>. Zugegriffen: 10.2.2019.
- [7] Offizielle Seite des eslint Linters. <https://eslint.org/>. Zugegriffen: 10.2.2019.
- [8] Offizielle Seite des PMD Werkzeugs für statische Code Analyse. <https://pmd.github.io/>. Zugegriffen: 10.2.2019.
- [9] Offizielle Seite des SpotBugs Werkzeugs für statische Code Analyse. <https://spotbugs.github.io/>. Zugegriffen: 10.2.2019.
- [10] Offizielle Seite des Google Java Style Guide. <https://google.github.io/styleguide/javaguide.html>. Zugegriffen: 8.2.2019.

A Anhang

Der Anhang lässt sich mithilfe des Inhaltsverzeichnisses navigieren, da jedoch einige der angehängten Dokumente einen eigenen Index mit eigener Seitennummerierung pflegen, wird die globale Seitenzahl des Dokuments, wie im Hauptindex angezeigt, ab sofort unten links in einer blauen Box zur Unterscheidung angezeigt.

Erste Nutzerstudie

Es folgen in dieser Sektion zunächst die zur Umsetzung der Studie eingesetzten Materialien:

- Aufgabenstellungen, die die Probanden bearbeitet haben (ab Sektion A.1.1).
- Evaluierungsbögen, die die Probanden nach der Bearbeitung ausfüllen sollten (ab Sektion A.1.3).
- Die Auswertung der Ergebnisse der Studie (ab Sektion A.1.5).

Weiterhin sind die ausgefüllten Evaluationsbögen in Sektion A.8.1 zu finden.

Die Studie wurde im Zeitraum vom 6. bis zum 11. Februar durchgeführt. Die Durchführung erfolgte durch die Mitglieder des Entwicklungsteams, welche, wie im Qualitätssicherungsdokument beschrieben, als Betreuer agiert haben.

Es handelte sich bei den Durchführenden also um:

- David Heck
- Jonas Belouadi
- Martin Kerscher
- Marc Arnold
- Anton Haubner

Teilgenommen haben 16 Studierende, die das Vorlesung FMiSE hören, oder bereits gehört haben, sowie 5 wissenschaftliche Mitarbeiter der Software Engineering Group der TU Darmstadt. Insgesamt gab es also 21 Teilnehmer an der Studie.

Nutzerstudie KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Aufgaben zur Evaluierung der Nutzbarkeit KollaborierbaRs

Projektverwaltung

In dieser Aufgabe soll es darum gehen, die grundlegenden Projektverwaltungsoperationen KollaborierbaRs auf ihre Benutzbarkeit zu testen. Im folgenden sollen Sie Projektstrukturen erzeugen, verändern und löschen:

- a) Bitte nutzen Sie KollaborierbaRs Projektverwaltung um ein Projekt der folgenden Struktur zu erzeugen:

```
projektverwaltung_TestGruppe_x
└── res
    ├── res1.java
    └── res2.java
└── src
    ├── Main.java
    └── models
        ├── Car.java
        └── Door.java
```

- b) Nehmen Sie nun eine Umstrukturierung des erstellen Projekts vor. Das Projekt soll danach die folgende Struktur haben:

```
projektverwaltung_TestGruppe_x
└── res
    ├── libary1
    │   └── Init.java
    └── libary2
        └── Init_Main.java
└── src
    └── Main.java
```

- c) Löschen Sie bitte das Projekt aus Aufgabe b).

KeY

In dieser Aufgabe sollen Sie die bisher in KollaborierbaR integrierten KeY Funktionen testen.

- Öffnen Sie mittels der Projektverwaltung KollaborierbaRs das Projekt 'Spezifikation_Testgruppe_x'. 'x' ist hierbei durch die Ihnen zugewiesene Gruppennummer zu substituieren.
- Öffnen Sie nun die Datei 'Door.java' im Ordner '/src/Car'. Dort finden Sie mehrere Methoden mit JML Spezifikationen. Starten Sie die Verifikation der Methode 'setColour(String colour)'. Interpretieren Sie anhand KollaborierbaRs Feedback, ob Methode verifiziert werden konnte.
- Öffnen Sie nun die Datei 'Car.java' im Ordner '/src/Car'. Auch hier finden Sie mehrere Methoden mit JML Spezifikationen. Starten Sie die Verifikation für alle Proof Obligation auf einmal. Interpretieren Sie anhand KollaborierbaRs Feedback ob alle Methoden verifiziert werden können.

Kollaboratives Editieren

In dieser Aufgabe sollte Sie gemeinsam mit Ihrer Gruppe die kollaborativen Funktionen testen. Öffnen Sie dafür bitte das Projekt 'kollaborative_Testgruppe_x'. Hier bei ist x durch die Ihnen zugewiesene Gruppennummer zu substituieren. Bitte editieren Sie die Datei '/src/Station.java', sodass der Inhalt dem beigelegten Quellcode gleicht.

Anhang

```
1 public class Station{  
2  
3     public String name;  
4     public String location;  
5     public int employeesCount;  
6  
7     public Station(String name, String location, int employeesCount){  
8         this.name = name;  
9         this.location = location;  
10        this.employeesCount = employeesCount;  
11    }  
12  
13    public String getName(){  
14        return this.name;  
15    }  
16  
17    public void setName(String name){  
18        this.name = name;  
19    }  
20  
21    public String getLocation(){  
22        return this.location;  
23    }  
24  
25    public void setLocation(String location){  
26        this.location = location;  
27    }  
28  
29    public int getEmployeesCount(){  
30        return this.employeesCount;  
31    }  
32  
33    public void setEmployeesCount(int employeesCount){  
34        this.employeesCount = employeesCount;  
35    }  
36}
```

Evaluierung

Fast geschafft! Nun müssen Sie nur noch Ihre Erfahrungen in unserem Fragebogen festhalten: <https://bit.ly/2Dhd5ii>.

Usability Study KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Tasks aimed at evaluating KollaborierbaRs usability

Project Management

The goal of this task is to check the usability of core project management features of KollaborierbaR. Your task is to create, modify and delete files and folders in a project.

- Please use KollaborierbaR to create a project with the following structure:

```
projektverwaltung_TestGruppe_x
├── res
│   ├── res1.java
│   └── res2.java
└── src
    ├── Main.java
    └── models
        ├── Car.java
        └── Door.java
```

- Please modify the project such that it conforms to the following structure:

```
projektverwaltung_TestGruppe_x
├── res
│   ├── library1
│   │   └── Init.java
│   └── library2
│       └── Init_Main.java
└── src
    └── Main.java
```

- Please delete the project you just created in task b).

KeY

In this task you can test the features of KeY that have already been implemented into KollaborierbaR.

- Open the projekt 'Spezifikation_Testgruppe_x'. Replace 'x' with your assigned group number.
- Now open the file 'Door.java' from the folder '/src/Car'. There you will find a few methods with their corresponding JML specifications. Verify the contracts of the method 'setColour(String colour)'. Conclude from the response on your screen whether the method could be proven.
- Please open the file 'Car.java' from the folder '/src/Car' which contains additional JML specifications. Trigger the verification for all proof obligations at once. Interpret whether all contracts could be proven.

Collaborative editing

For this task we ask you to test the collaborative features together with your group. Therefore please open the project 'kollaborative_Testgruppe_x'. Replace 'x' with your assigned group number. Please edit the file '/src/Station.java', to match the attached source code.

Anhang

```
1  public class Station{  
2  
3      public String name;  
4      public String location;  
5      public int employeesCount;  
6  
7      public Station(String name, String location, int employeesCount){  
8          this.name = name;  
9          this.location = location;  
10         this.employeesCount = employeesCount;  
11     }  
12  
13     public String getName(){  
14         return this.name;  
15     }  
16  
17     public void setName(String name){  
18         this.name = name;  
19     }  
20  
21     public String getLocation(){  
22         return this.location;  
23     }  
24  
25     public void setLocation(String location){  
26         this.location = location;  
27     }  
28  
29     public int getEmployeesCount(){  
30         return this.employeesCount;  
31     }  
32  
33     public void setEmployeesCount(int employeesCount){  
34         this.employeesCount = employeesCount;  
35     }  
36 }
```

Evaluation

You're almost done! Now you only need to evaluate the user experience in our evaluation form: <https://tinyurl.com/y8yohdh>

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

* Erforderlich

Vorwissen

1. Vorwissen *

Markieren Sie nur ein Oval.

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
- Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY

Weiter mit Frage 2

Projektverwaltung

2. Gab es beim Erstellen von Projekten Probleme? *

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

3. Anmerkung:

4. Gab es beim Erstellen von Dateien und Ordnern Probleme?

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

5. Anmerkung:

6. Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

7. Anmerkung:

8. Gab es beim Öffnen von Projekten und Dateien Probleme?

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

9. Anmerkung:

KeY

10. Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweise zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

11. Anmerkung:

12. Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

Markieren Sie nur ein Oval.

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Kollaboratives Editieren

13. Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

Markieren Sie nur ein Oval.

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

14. Anmerkung:

15. Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

Markieren Sie nur ein Oval.

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
 - Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
 - Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

16. Anmerkung:

Sonstige Anmerkungen

17.

Bereitgestellt von



Evaluation of KollaborierbaRs usability

You're almost done. Now all that is left is to record your experience in this questionnaire. We ask you to answer all questions honestly.

* Erforderlich

Prior knowledge

1. *

Markieren Sie nur ein Oval.

- I am a student and I am attending the lecture "Formale Methoden im Software Entwurf"/"Formal Methods in Software Engineering" or have done so in the past.
- I am an employee at TU Darmstadt and I am working with KeY or on improving KeY.

Weiter mit Frage 2

Editing projects

2. Did you encounter problems while creating projects? *

Markieren Sie nur ein Oval.

- I had no problems creating projects.
- I had to get familiar with the interface briefly, but then I was able to easily create projects.
- I was able to create projects, but only with the help of a supervisor.
- I was not able to create projects.

3. Remarks:

4. Did you encounter difficulties while creating files or folders?

Markieren Sie nur ein Oval.

- I had no problems creating files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily create files and folders.
- I was able to create files and folders, but only with the help of a supervisor.
- I was not able to create files or folders.

5. Remarks:

6. Did you encounter difficulties while deleting projects, files or folders?

Markieren Sie nur ein Oval.

- I had no problems deleting files, folders, or projects.
- I had to get familiar with the interface briefly, but then I was able to easily delete projects, files or folders.
- I could delete projects, folders, and files only with the help of a supervisor.
- I was not able to delete files or folders.

7. Remarks:

8. Did you encounter difficulties while opening projects, files or folders?

Markieren Sie nur ein Oval.

- I had no problems opening projects, files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily open projects, files or folders.
- I could open projects, folders, or files only with the help of a supervisor.
- I was not able to open files or folders.

9. Remarks:

KeY

10. Did you encounter problems when proving method contracts individually?

Markieren Sie nur ein Oval.

- I experienced no difficulties proving method contracts individually.
- Only after identifying the differences to the "old" KeY interface, I was able to prove methods individually.
- I could prove individual methods only with the help of a supervisor.
- I was not able to prove methods individually.

11. Remarks:

12. Was it difficult to evaluate whether a proof was successful?

Markieren Sie nur ein Oval.

- KollaborierbaRs UI made it clear to me, whether a proof was successful or not.
- I could tell if a proof was successful only after consulting a supervisor.
- I was not able to tell, whether a proof completed successfully or not.

Collaborative Editing

13. Did you notice inconsistent states when editing files collaboratively?

Markieren Sie nur ein Oval.

- I did not notice any inconsistencies while editing.
- From time to time I noticed an inconsistency. However I was able to correct it by reloading the file.
- Collaborative editing did not work for me.

14. Remarks:

**15. Code sections written by other users have been highlighted by the editor.
Did this support you while collaboratively editing files?**

Markieren Sie nur ein Oval.

- Yes, it did help me to get an overview of changes to the file.
- It neither supported me nor did it have a negative impact.
- No, it distracted me from the task.

16. Remarks:

Additional remarks:

Bereitgestellt von



Auswertung der 1. Nutzerstudie - KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Inhaltsverzeichnis

1	Multiple-Choice Antworten	2
1.1	Vorwissen	2
1.2	Projektverwaltung	3
1.3	KeY	3
1.4	Kollaboratives Editieren	4
1.5	Beobachtungen	4
2	Nutzeranmerkungen aus Freitextfeldern und Betreuernotizen	5
2.1	Zusammenfassung der Betreuernotizen und dem Freitextfeedback	5
2.2	Besonders einfach zu implementierende Anmerkungen	7
2.3	Eher schwierig zu implementierende Anmerkungen	8
3	Fazit	10
3.1	Projektverwaltung	10
3.2	KeY Integration	10
3.3	Kollaboratives Editieren	10

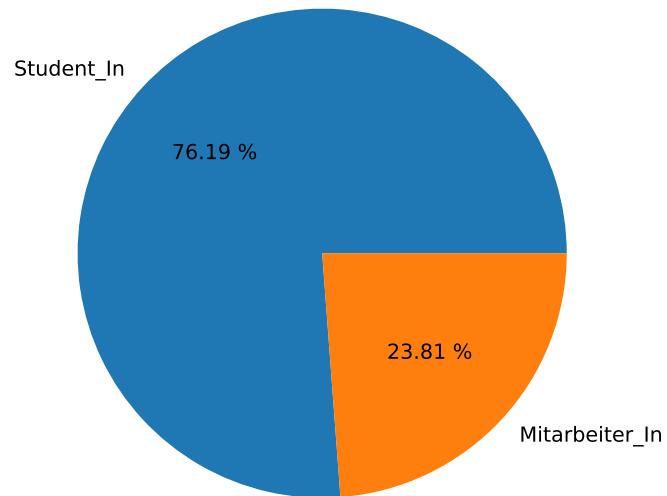
1 Multiple-Choice Antworten

Dieser Abschnitt fasst mit einigen Grafiken kurz zusammen, welchen allgemeinen Eindruck die Probanden von den einzelnen Bestandteilen der UI aufnahmen. Daher dienen die Daten der Multiple-Choice Fragen hier als Grundlage.

Zuletzt folgt noch eine kurze Zusammenfassung der eindeutigsten Beobachtungen in diesem Abschnitt.

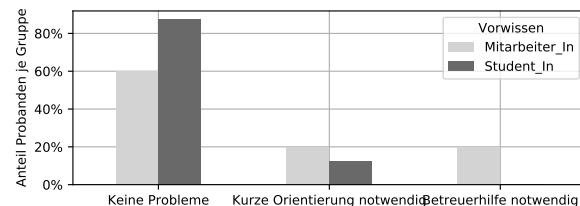
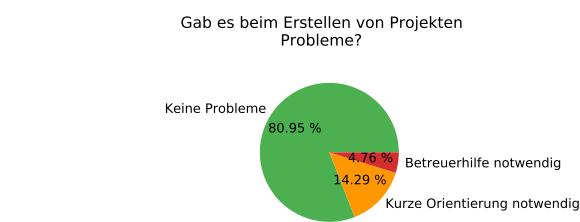
1.1 Vorwissen

Anteile der Zielgruppen an den Probanden

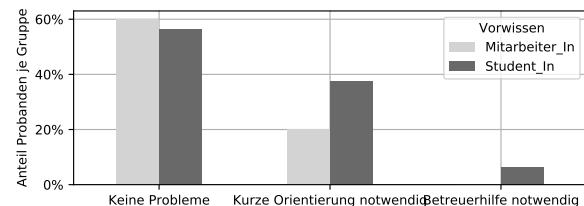
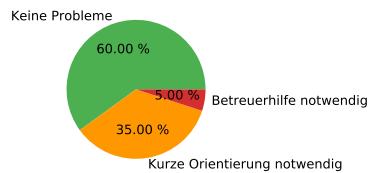


An der Studie haben 21 Personen teilgenommen, davon 16 Studierende und 5 Mitarbeiter_Innen.

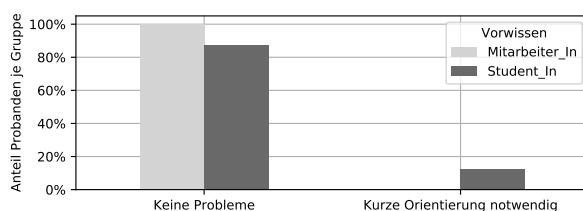
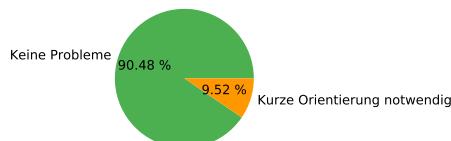
1.2 Projektverwaltung



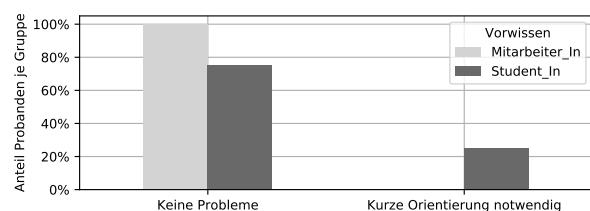
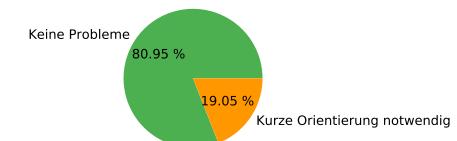
Gab es beim Erstellen von Dateien und Ordnern Probleme?



Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

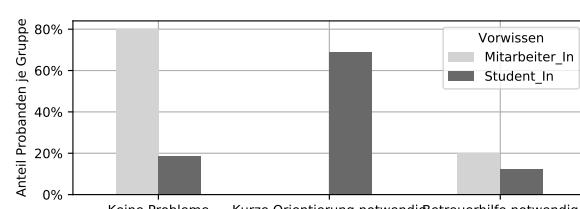
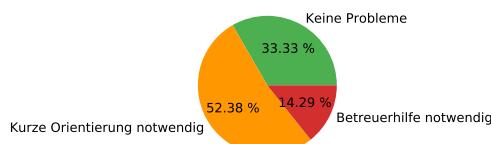


Gab es beim Öffnen von Projekten und Dateien Probleme?

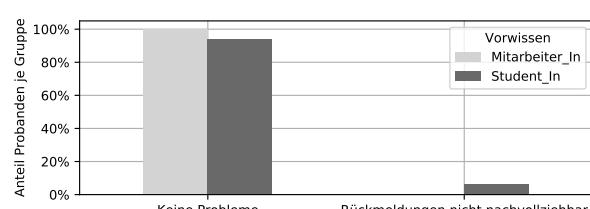
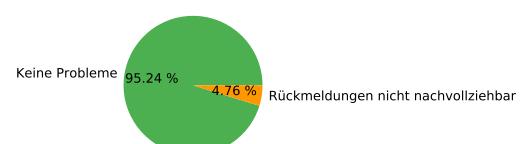


1.3 KeY

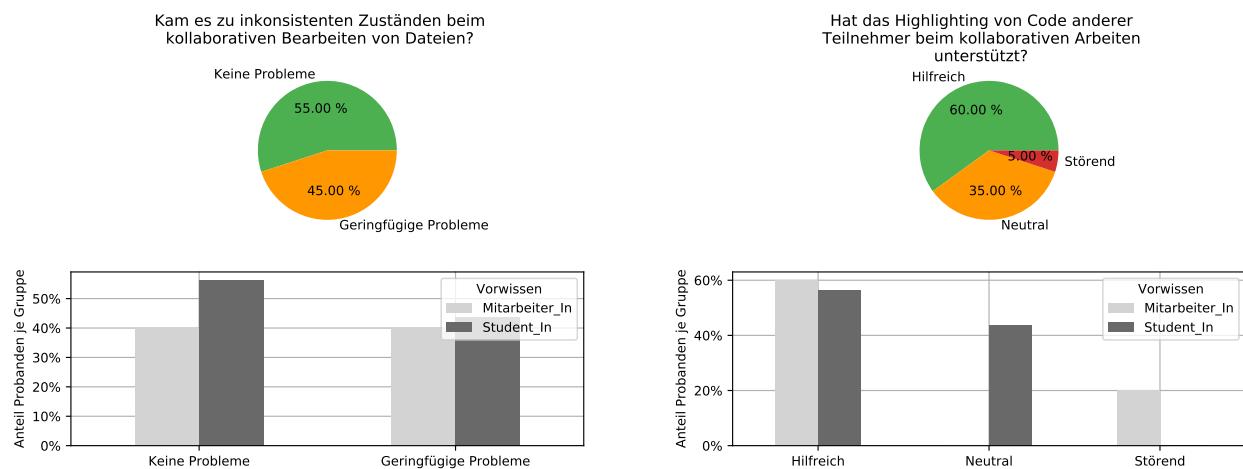
Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?



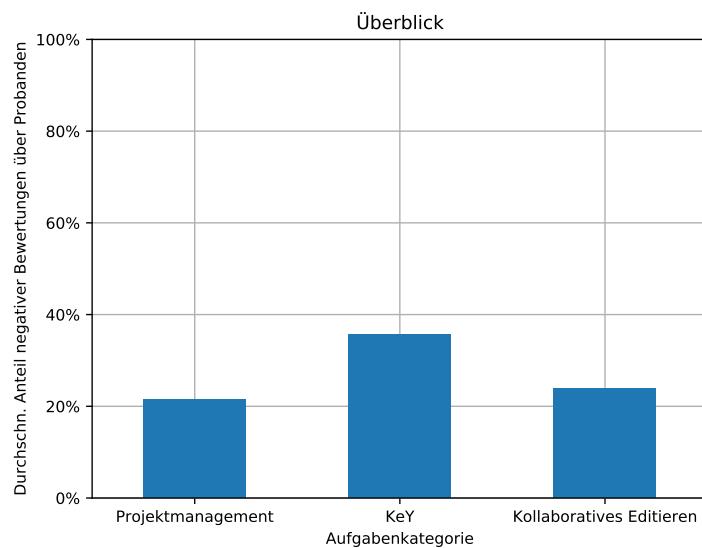
Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?



1.4 Kollaboratives Editieren



1.5 Beobachtungen



- Das Projektmanagement wurde am positivsten durch die Probanden aufgefasst
- Die KeY Integration wurde im Durchschnitt eher negativ aufgefasst.
- Im Durchschnitt wurde keiner der Bereiche in über 36 % der Fragen negativ bewertet.
- In den Bereichen Projektmanagement und KeY kamen die wiss. Mitarbeiter besser zurecht als die Studierenden, allerdings haben sie wieder etwas mehr Kritik am kollaborativen Arbeiten deutlich gemacht
- Beim Projektmanagement mussten sich die Probanden vor allem beim Erstellen von Dateien und Ordnern zunächst orientieren. Auch bei KeY hatten die Probanden zunächst Zeit benötigt, um einzelne Beweise mit der UI durchführen zu können. Fast die Hälfte jeweils der Mitarbeiter und Studierenden erfuhren in gleichem Maße geringfügige Probleme beim kollaborativen Editieren.

2 Nutzeranmerkungen aus Freitextfeldern und Betreuernotizen

Die Daten der Multiple-Choice Fragen haben vor allem ein allgemeines Stimmungsbild erfasst und erste Eindrücke geschaffen, wie gut die verschiedenen Zielgruppen (Mitarbeiter / Studierende) jeweils mit der Benutzeroberfläche zurechtgekommen sind.

Neben den Multiple-Choice Fragen wurde aber auch direktes Feedback der Probanden in Form von z. B. Verbesserungsvorschlägen und entdeckten Fehlern erfasst. Dies erfolgte über Freitextfelder im Fragebogen und durch Notizen des jeweiligen Betreuers des Probanden, welcher auch Feedback im direkten Gespräch erfasst hat.

Diese Daten können nun dabei helfen, die Hintergründe des Stimmungsbilds aus den Multiple-Choice Fragen aufzuzeigen. Weiterhin wird die Verbesserung der Benutzbarkeit KollaborierbaRs effektiv durch die geäußerten Verbesserungsvorschläge unterstützt, welche nun evaluiert und ggf. realisiert werden können. Es folgt zunächst eine Übersicht allen Feedbacks und dann eine Kategorisierung der Verbesserungsvorschläge je nach ihrer Umsetzbarkeit.

2.1 Zusammenfassung der Betreuernotizen und dem Freitextfeedback

Die folgende Zusammenfassung listet alle durch die Probanden genannten Anmerkungen. Wurde eine Thematik mehrfach genannt, so wurde das Feedback zusammengefasst und notiert, von wie vielen Probanden eine vergleichbare Anmerkung geäußert wurde (wie bezeichnen diese Anzahl im Folgenden als *Priorität*).

Dies soll es erlauben zu gewichten, welche Verbesserungen sich von den Benutzern besonders gewünscht werden:

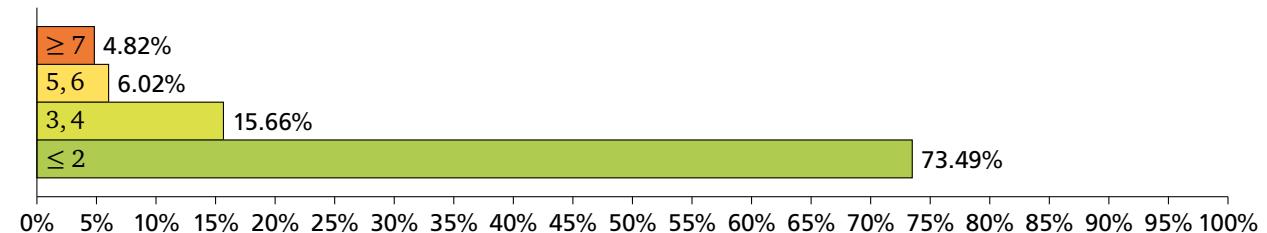
Nr.	Anmerkung	Anzahl Vorkommnisse
1	Die visuelle Einrückung von geöffneten Dateien verwirrt	9
2	Button für Einzelbeweise nicht oder nur schwer gefunden.	9
3	Rechtsklick auf Methoden zum Ausführen von Beweisen	8
4	Versucht Datei durch Ziehen zu verschieben	8
5	Bitte Sortierung (alphabetisch und nach Ordnern/Dateien) der Sidebar	6
6	Doppelklick für Projekte öffnen	5
7	Download von Dateien	5
8	Strg+Z wie gewohnt, soll nicht Änderungen anderer Personen betreffen	5
9	Bitte (alphabetische) Sortierung der Projektauswahl	5
10	Kontextmenü der Sidebar: Die Reihenfolge der Optionen („delete“ vor „create“) sei verwirrend, lieber an bestehenden IDEs orientieren	4
11	Einfacher Klick statt Doppelklick zum Öffnen von Dateien	4
12	Wunsch: Der Cursor der anderen Mitbearbeiter soll angezeigt werden	4
13	Prompts sollen denjenigen Button hervorheben, welche beim Bestätigen mit Enter automatisch ausgelöst werden	4
14	Farben der Personen sind verwirrend (Farbe je nach geöffneter Datei), lieber eine Farbe für jede Person im ganzen Projekt	4
15	Shortcuts für Vervollständigung von Code gewünscht	3
16	Verbesserung: Symbole im Kontextmenü der Sidebar	3
17	Menüpunkt gewünscht: File -> Create Folder/File	3
18	Das Highlighting ist zu unauffällig, man bemerkt es zunächst nicht	3
19	Möchte Suchfunktion für Projekte	3
20	Verbesserung: Ordner werden mit einem Klick geöffnet, Dateien mit zwei, das sei inkonsistent	3
21	Scrollen beim Schreiben von Code	3
22	„Open Goals“ sollten sich scrollen lassen, oder in der Zeile umbrechen	3
23	Wunsch: Ein neues Projekt solle sich per Rechtsklick auch in der Sidebar erstellen lassen	2
24	Umbenennen von Projekten	2
25	Bug: Upload von Dateien sollte gehen, funktioniert aber nur bei geschlossenem Projekt	2
26	Will auf die Goals klicken	2
27	Wunsch: Automatische Codevervollständigung	2

28	Wunsch: Man sollte das Theme ändern können (z. B. Light-Theme)	2
29	Wunsch: Man sollte sehen können, was andere Personen markiert haben	2
30	Wunsch: Mehr Details zum (Miss-)Erfolg eines Beweises	2
31	Markieren ausschalten für nicht benötigte Elemente in der UI	2
32	Nach Öffnen eines anderen Projekts stehen bei Open Goals noch immer die Goals vom letzten Projekt	2
33	„Prove all contracts“ vielleicht als eigenen Button, sei ein wenig zu versteckt	2
34	Verbesserung: Man sollte Dateien/Ordner auch in der Sidebar erstellen können, wenn man ins Leere klickt	2
35	Bug: Projekte schließen sich manchmal, wenn man Ordner gleichzeitig löscht	2
36	Es sei nicht offensichtlich, dass Kontextmenüs sich per Rechtsklick öffnen lassen. Ein Hinweis wäre hilfreich	2
37	Kollaborierbares Editieren sollte in Textblöcken erfolgen, nicht in einzelnen Buchstaben, sei natürlicher und würde Edits weniger vermischen	2
38	Wunsch: Eine Übersicht von Klassen und Methoden (á la IntelliJ) über die man auch Beweise starten kann	2
39	Sidebar reagiert nicht	1
40	Fehlerfenster zu klein	1
41	Manuelles Speichern vor dem Beweisen stört	1
42	Racecondition bei gleichzeitigem Löschen	1
43	Neuladen der Seite sollte Projekt nicht schließen	1
44	Projektname sieht zu sehr aus wie ein Ordner	1
45	Ein Klick auf das Logo oben links führt zum Schließen des Projekts	1
46	Emojis in Dateinamen/Ordnernamen sollten nicht möglich sein	1
47	Projekte sollten mit Tastatur auswählbar sein	1
48	Das Umbenennen einer Datei zu bestehendem Namen führe zum Verschwinden der Datei	1
49	Semantischer Fehler in Notifications: „Proving obligations“ statt „Running proof obligations“	1
50	Die Farben verschiedener Personen sollten unbedingt verschieden sein	1
51	Eingebauter Chat gewünscht	1
52	Menüpunkt gewünscht: File -> Delete Folder/File	1
53	Wunsch: In Projekt auswahl zuletzt geöffnete Projekte anzeigen.	1
54	„Prove all contracts“ sollte man auch oben im Gutter auswählen können	1
55	Wenn ein Beweis fehlschlägt, sollte das durch besonderes Symbol im Gutter gezeigt werden (z.B. rotes X)	1
56	Wunsch: Wenn ein Beweis aufgrund nicht ausreichender Suchtiefe fehlschlägt soll ein Fortsetzen mit erhöhter Tiefe möglich sein, am besten mit einem „Resume“ Symbol im Gutter	1
57	Wunsch: Man sollte das ganze Projekt herunterladen können	1
58	Wunsch: Man sollte ein „Lock“ auf Dateien setzen können, sodass nur 1 Person es bearbeiten kann	1
59	Wunsch: Auch Ergebnisse von Beweisen sollen geteilt werden können	1
60	Bug: Ordner würden sich manchmal ohne Grund schließen	1
61	Wunsch: Offene Dateien sollten als Tabs dargestellt werden	1
62	Bug: Schließt man ein Projekt, könne man es erst nach Neuladen der Seite wieder öffnen	1
63	Verbesserung: Das Y in KeY bitte konsistent groß schreiben	1
64	Bug: Die Rückmeldungen zu den Beweisen zeigen immer „contract-0“ an, auch wenn es mehrere Kontrakte gab	1
65	Verbesserung: Beweisknöpfe im Gutter sollten heller sein um sie leichter zu finden	1
66	Wunsch: Man sollte Beweise einzelner Kontrakte auch über die Menüleiste finden können	1
67	Bug: Der Cursor sei manchmal mit dem des anderen Bearbeiters identisch, bis man manuell an eine andere Stelle klickt	1
68	Bug: Bei langer Bearbeitung einer Datei durch viele Teilnehmer falle die Datei out of sync, bis ein neuer Teilnehmer beitritt. Auch Neuladen der Seite sei in diesem Fall ohne Wirkung	1

69	Verbesserung: Mehr und bunte Farben	1
70	Bug: Das Highlighting der Eingaben anderer wurde nur teilweise angezeigt	1
71	Wunsch: Ein persistenter Verlauf der Änderungen à la git blame	1
72	Wunsch: Nach dem automatischen Erstellen einer geschlossenen Klammer sollte die Eingabe einer geschlossenen Klammer keine weitere Klammer erzeugen und die bestehende überspringen	1
73	Wunsch: Nach dem Dateiende sollte man trotzdem weiterscrollen können	1
74	Wunsch: Upload per Drag-and-drop	1
75	Wunsch: Man sollte alle Notifications auf einmal löschen können mit einem Klick	1
76	Wunsch: Automatische Formatierung (durch Tastenkombination)	1
77	Verbesserung: Das Java-Highlighting nutze die gleiche Farbe für zu viele Elemente und sollte diverser sein	1
78	Wunsch: Man sollte mehrere Projekte auf einmal öffnen können	1
79	Wunsch: „About“-Page mit Erklärung, worum es sich bei KollaborierbaR überhaupt handelt	1
80	Wunsch: Man sollte „Open-Goals“ optional auch unter der Projektstruktur anzeigen können, statt Tabs	1
81	Wunsch: Man sollte ein Programm auch ausführen können	1
82	Wunsch: Tastenkombinationen für wichtigste Funktionen sollten eingebaut werden und bei den zugehörigen Schaltflächen angezeigt werden	1
83	Verbesserung: Man sollte Textlabels im Interface nicht markieren können, da es sonst zu sehr wie eine Website wirkt	1

Insgesamt wurden also 83 Anmerkungen erfasst...

- davon 4 mit Priorität **größer gleich 7** (4,82 %)
- davon 5 mit Priorität **5 oder 6** (6,02 %)
- davon 13 mit Priorität **3 oder 4** (15,66 %)
- davon 61 mit Priorität **kleiner oder gleich 2** (73,49 %)



2.2 Besonders einfach zu implementierende Anmerkungen

Nr.	Anmerkung	Anzahl Vorkommnisse
1	Die visuelle Einrückung von geöffneten Dateien verwirrt	9
5	Bitte Sortierung (alphabetisch und nach Ordnern/Dateien) der Sidebar	6
6	Doppelklick für Projekte öffnen	5
9	Bitte (alphabetische) Sortierung der Projektauswahl	5
10	Kontextmenü der Sidebar: Die Reihenfolge der Optionen („delete“ vor „create“) sei verwirrend, lieber an bestehenden IDEs orientieren	4
11	Einfacher Klick statt Doppelklick zum Öffnen von Dateien	4
23	Wunsch: Ein neues Projekt solle sich per Rechtsklick auch in der Sidebar erstellen lassen	2
32	Nach Öffnen eines anderen Projekts stehen bei Open Goals noch immer die Goals vom letzten Projekt	2
33	„Prove all contracts“ vielleicht als eigenen Button, sei ein wenig zu versteckt	2
34	Verbesserung: Man sollte Dateien/Ordner auch in der Sidebar erstellen können, wenn man ins Leere klickt	2

40	Fehlerfenster zu klein	1
41	Manuelles Speichern vor dem Beweisen stört	1
45	Ein Klick auf das Logo oben links führt zum Schließen des Projekts	1
49	Semantischer Fehler in Notifications: „Proving obligations“ statt „Running proof obligations“	1
52	Menüpunkt gewünscht: File -> Delete Folder/File	1
55	Wenn ein Beweis fehlschlägt, sollte das durch besonderes Symbol im Gutter gezeigt werden (z.B. rotes X)	1
63	Verbesserung: Das Y in KeY bitte konsistent groß schreiben	1
65	Verbesserung: Beweisknöpfe im Gutter sollten heller sein um sie leichter zu finden	1
75	Wunsch: Man sollte alle Notifications auf einmal löschen können mit einem Klick	1
79	Wunsch: „About“-Page mit Erklärung, worum es sich bei KollaborierbaR überhaupt handelt	1
83	Verbesserung: Man sollte Textlabels im Interface nicht markieren können, da es sonst zu sehr wie eine Website wirkt	1

- davon 1 mit Priorität **größer gleich 7**
- davon 3 mit Priorität **5 oder 6**
- davon 2 mit Priorität **3 oder 4**
- davon 15 mit Priorität **kleiner oder gleich 2**

Insgesamt ist also ein Anteil von 25,30 % an den Gesamtanmerkungen nach unserer Einschätzung relativ einfach zu beheben.

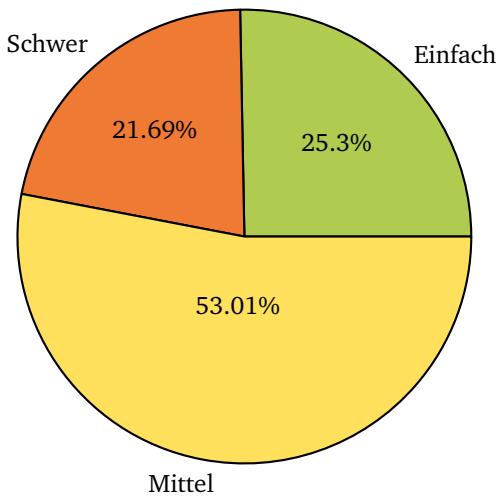
2.3 Eher schwierig zu implementierende Anmerkungen

Nr.	Anmerkung	Anzahl Vorkommnisse
8	Strg+Z wie gewohnt, soll nicht Änderungen anderer Personen betreffen	5
12	Wunsch: Der Cursor der anderen Mitbearbeiter soll angezeigt werden	4
15	Shortcuts für Vervollständigung von Code gewünscht	3
27	Wunsch: Automatische Codevervollständigung	2
29	Wunsch: Man sollte sehen können, was andere Personen markiert haben	2
35	Bug: Projekte schließen sich manchmal, wenn man Ordner gleichzeitig löscht	2
37	Kollaborierbares Editieren sollte in Textblöcken erfolgen, nicht in einzelnen Buchstaben, sei natürlicher und würde Edits weniger vermischen	2
38	Wunsch: Eine Übersicht von Klassen und Methoden (á la IntelliJ) über die man auch Beweise starten kann	2
42	Racecondition bei gleichzeitigem Löschen	1
51	Eingebauter Chat gewünscht	1
58	Wunsch: Man sollte ein „Lock“ auf Dateien setzen können, sodass nur 1 Person es bearbeiten kann	1
59	Wunsch: Auch Ergebnisse von Beweisen sollen geteilt werden können	1
67	Bug: Der Cursor sei manchmal mit dem des anderen Bearbeiters identisch, bis man manuell an eine andere Stelle klickt	1
68	Bug: Bei langer Bearbeitung einer Datei durch viele Teilnehmer falle die Datei out of sync, bis ein neuer Teilnehmer beitritt. Auch Neuladen der Seite sei in diesem Fall ohne Wirkung	1
71	Wunsch: Ein persistenter Verlauf der Änderungen á la git blame	1
76	Wunsch: Automatische Formatierung (durch Tastenkombination)	1
78	Wunsch: Man sollte mehrere Projekte auf einmal öffnen können	1
81	Wunsch: Man sollte ein Programm auch ausführen können	1

- davon 0 mit Priorität **größer gleich 7**
- davon 1 mit Priorität **5 oder 6**

- davon 2 mit Priorität **3 oder 4**
- davon 15 mit Priorität **kleiner oder gleich 2**

Insgesamt ist also ein Anteil von 21,69 % an den Gesamtanmerkungen nach unserer Beurteilung nur vergleichsweise schwierig zu beheben.



3 Fazit

Durch die Nutzerstudie gingen einige Schwachstellen von KollaborierbaR hervor, die einer Verbesserung bedürfen. Wir gehen an dieser Stelle noch einmal auf die drei Themengebiete „Projektverwaltung“, „KeY Integration“ und „Kollaboratives Editieren“ ein und stellen unser geplantes weiteres Vorgehen vor.

3.1 Projektverwaltung

Hier wurde mittels der Multiple-Choice Antworten die höchste Zufriedenheit im Vergleich zu den anderen Themengebieten festgestellt. Die schlechtesten Bewertungen erhielt die Funktionalität zum Erstellen von Dateien und Ordnern, allerdings auch hier haben nur 5% der Teilnehmer angegeben, dass sie Betreuerhilfe benötigt haben, und 35% mussten sich lediglich kurz orientieren.

Für die Umsetzung von Maßnahmen haben wir in Absprache mit dem Auftraggeber versucht, eine Balance zwischen Komplexität, Dringlichkeit und neuen Features zu finden. Die Anmerkungen 1, 5, 6, 9 und 10 haben wir zur Bearbeitung in Userstories gefasst:

- Anpassung der Sortierung der Operationen in der Menüführung (ASOM1402) als Verbesserung für Anmerkung Nr. 10
- Sortierung von Projekten (SvP2502) als Verbesserung für Anmerkung Nr. 9
- Einheitliche Einrückung der Dateistruktur (EedD1402) als Verbesserung für Anmerkung Nr. 1
- Doppelklick auf Projektnamen öffnet Projekte (DaP1402) als Verbesserung für Anmerkung Nr. 6
- Sortierung der Sidebar (SsS2502) als Verbesserung für Anmerkung Nr. 5

3.2 KeY Integration

Da die KeY Integration im Durchschnitt am negativsten aufgenommen wurde, hat die Verbesserung dieser eine besonders hohe Priorität. Hier meldeten 52% der Teilnehmer, dass eine kurze Orientierung notwendig gewesen sei. 14.29% benötigten Hilfe durch den Betreuer. Wir führen dieses Ergebnis auf eine unintuitive Integration von KeY zurück.

Den Multiple-Choice Antworten zu KeY ließ sich entnehmen, dass das Starten von Beweisen zu einzelnen Methodenverträgen in einem Großteil der Fälle zu Orientierungsproblemen führte. Im Einklang dazu ging aus Anmerkung 3 hervor, dass eine Rechtsklickfunktion auf Methoden zur Interaktion mit KeY häufig vermisst wurde. Deshalb wurde gemeinsam mit dem Auftraggeber beschlossen dieses fehlende Feature auf jeden Fall zu implementieren.

Zusätzlich zu Anmerkung 3 wurden von uns noch 22 und 41 zur weiteren Bearbeitung in Userstories gefasst:

- Rechtsklick auf Methoden zum Beweisen von Proof Obligation (RaMzBvP2502) als Verbesserung für Anmerkung Nr. 3
- Scrollen innerhalb der Anzeige von Open Goals (SidAvOG2502) als Verbesserung für Anmerkung Nr. 22
- Automatisches Speichern vor dem Starten von Beweisen (AsvdSvB2502) als Verbesserung für Anmerkung Nr. 41

3.3 Kollaboratives Editieren

45% der Teilnehmer stellten geringfügige Probleme beim kollaborativen Arbeiten fest, z. B. inkonsistente Texte zwischen den verschiedenen Bearbeitern einer Datei. 5% empfanden das Highlighting der Änderungen anderer Teilnehmer als störend.

Die Anmerkungen 8, 21 und 27 wurden von uns zur weiteren Bearbeitung in Userstories gefasst:

- Rückgängig durch Steuerung Z (RdSZ1402) als Verbesserung für Anmerkung Nr. 8
- Automatisches Scrollen beim Schreiben von Code (AsbSvC2502) als Verbesserung für Anmerkung Nr. 21
- Auto vervollständigung beim Editieren (AbE1402) als Verbesserung für Anmerkung Nr. 27

Zweite Nutzerstudie

Es folgen in dieser Sektion zunächst die zur Umsetzung der Studie eingesetzten Materialien:

- Aufgabenstellungen, die die Probanden bearbeitet haben (ab Sektion A.2.1).
- Evaluierungsbögen, die die Probanden nach der Bearbeitung ausfüllen sollten (ab Sektion A.2.3).
- Die Auswertung der Ergebnisse der Studie (ab Sektion A.2.5).

Weiterhin sind die ausgefüllten Evaluationsbögen in Sektion A.8.1.3 zu finden.

Die Studie wurde im Zeitraum vom 11. bis zum 15. März durchgeführt. Die Durchführung erfolgte durch die Mitglieder des Entwicklungsteams, welche, wie im Qualitätssicherungsdokument beschrieben, als Betreuer agiert haben.

Es handelte sich bei den Durchführenden also um:

- David Heck
- Jonas Belouadi
- Martin Kerscher
- Marc Arnold
- Anton Haubner

Teilgenommen haben 13 Studierende, die das Vorlesung FMiSE hören, oder bereits gehört haben, sowie 4 wissenschaftliche Mitarbeiter der Software Engineering Group der TU Darmstadt. Insgesamt gab es also 17 Teilnehmer an der Studie.

2. Nutzerstudie KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Aufgaben zur Evaluierung der Nutzbarkeit KollaborierbaRs

Aufgabe 1 Wiederholung: Projektmanagement

- a) Erstellen Sie ein Projekt 'projektvw_Gruppe_x' mit der folgenden Verzeichnisstruktur (x ist hierbei durch die Ihnen zugewiesene Gruppennummer zu substituieren):

```
projektvw_Gruppe_x
└── src
    ├── File1.java
    └── File2.java
```

- b) Löschen Sie als nächstes 'File1.java' und 'File2.java'. Löschen Sie dann den Ordner 'src'. Löschen Sie dann das gesamte Projekt.

Aufgabe 2 Fehlerhafte Spezifikation

Seit der letzten Nutzerstudie haben wir den KeY Beweisbaum und das Anzeigen des Sequenten in KollaborierbaR integriert. Damit sollen Sie nun nachvollziehen, weshalb das Beweisen eines Programms fehlschlägt und die gegebene Spezifikation verbessern.

Öffnen Sie dafür zuerst das Projekt 'Beweisen_Gruppe_x' (x ist hierbei durch die Ihnen zugewiesene Gruppennummer zu substituieren). In der Datei 'src/Car.java' finden Sie die Methode 'assignNewCarColor(Car c1, Car c2)' und eine dazugehörige JML-Spezifikation (vgl. Listing 1).

Starten Sie den Beweis der JML-Spezifikation und interpretieren Sie das Ergebnis. Suchen Sie anhand des Beweisbaums und der Anzeige des Sequenten das Problem und verbessern Sie die Spezifikation. Nehmen Sie keine Änderungen am Code vor.

Listing 1: Ausschnitt der Datei: Car.java

```
1  public class Car{
2
3      // Rest of the class
4
5      /*@ public normal_behaviour
6          @@
7          @ ensures c2.colour == \old(c2.colour);
8          @@
9          @*/
10     public static void assignNewCarColor(Car c1, Car c2){
11         c1.colour = "black";
12     }
13
14     // Rest of the class
15 }
```

Aufgabe 3 Suche im Array

In dieser Aufgabe soll Sie einen einfachen Suchalgorithmus auf Arrays implementieren und spezifizieren. Damit sollen alte und neue Features durch einen minimalen, realistischen Arbeitsablauf getestet werden. Bitte teilen Sie sich die Aufgaben b) und c) mit Ihrem Partner auf.

-
- a) Legen Sie ein Projekt mit der folgenden Struktur an:

```
ArraySuche_Gruppe_x
└── sry
    └── Main.java
```

- b) Implementieren Sie einen einfachen Suchalgorithmus auf Arrays mit der folgenden Signatur in der Datei 'Main.java':

```
1 public static boolean findElementInArray(int[] arr, int ele)
```

Der Algorithmus soll den Integer *ele* im gegebenen Array *arr* suchen. Falls das Element gefunden wird, soll true zurückgegeben werden. Falls das Element nicht gefunden wurde, soll false zurückgegeben werden.

Nutzen Sie zum Implementieren des Algorithmus eine *while*-Schleife und vermeiden Sie abrupte Terminierung innerhalb der Schleife (*breaks, continues, returns, exceptions*).

- c) Spezifizieren Sie den in b) beschriebenen Algorithmus. Folgende Aspekte sollten in ihrer Spezifikation enthalten sein:

- *arr* hat mindestens ein Element.
- Falls das Element *ele* im Array *arr* gefunden wurde, wird true zurückgegeben. Andernfalls soll false zurückgegeben werden.
- *findElementInArray(int[] arr, int ele)* ist seiteneffektfrei.

Des Weiteren müssen Sie die zur Umsetzung des Algorithmus benötigte Schleife spezifizieren. Spezifizieren Sie eine Schleifeninvariante (*loop invariant*), eine *assignable*-Klausel und einen *decreases*-Term.

- d) Beweisen Sie nun, dass ihr Suchalgorithmus ihre Spezifikation erfüllt. Sollte der Beweis scheitern, analysieren Sie das Problem anhand des Beweisbaums. Verbessern Sie dann Spezifikation und Code und starten Sie den Beweis erneut.

Evaluierung

Fast geschafft! Nun müssen Sie nur noch Ihre Erfahrungen in unserem Fragebogen festhalten: <https://bit.ly/2F27qOS>

2. User study KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Exercises for evaluating usability

Aufgabe 1 Rerun Project management

- a) Create a project 'projektvw_Gruppe_x' with the following folder structure (where x is your given group number):

```
projektvw_Gruppe_x
└── src
    ├── File1.java
    └── File2.java
```

- b) Next delete 'File1.java' and 'File2.java'. Furthermore delete the folder 'src'. To finish delete the whole project.

Aufgabe 2 Faulty specification

Since the last user study we have added the Key proof tree and the ability to display the Sequent to KollaborierbaR. With those extension you now need to retrace why a proof fails and correct the specification.

First open the project 'Beweisen_Gruppe_x' (where x is your given group number). In the file 'src/Car.java' you need to find the method 'assignNewCarColor(Car c1, Car c2)' and the corresponding JML-specification (see Listing 1).

Start with proving the JML-specification and interpret the result. Search, with help of the proof tree and the display of the Sequent where the problem is and improve the specification. You must not change the code.

Listing 1: Extract from the file: Car.java

```
1  public class Car{
2
3      // Rest of the class
4
5      /*@ public normal_behaviour
6          @@
7          @ ensures c2.colour == \old(c2.colour);
8          @@
9          @*/
10     public static void assignNewCarColor(Car c1, Car c2){
11         c1.colour = "black";
12     }
13
14     // Rest of the class
15 }
```

Aufgabe 3 Search inside of Arrays

In order to test old and new features with a minimal but realistic work flow, you need to implement and specify a simple search algorithm on arrays. Please split the exercise b) and c) with a partner.

- a) Create a project with the following structure:

```
ArraySuche_Gruppe_x
└── sry
    └── Main.java
```

-
- b) Implement a simple search algorithm on arrays with the following signature in the file 'Main.java':

```
1 public static boolean findElementInArray(int[] arr, int ele)
```

The algorithm should search for the Integer *ele* in the given Array *arr*. If the array contains the element return true, if not return false.

When implementing the algorithm use a *while*-loop and avoid abrupt termination inside of the loop (*breaks, continues, returns, exceptions*).

- c) Specify the algorithm as described in b). The following aspects should be contained in your specification:

- *arr* has at least one element.
- If the element *ele* is found in the array *arr*, return true. If not return false.
- *findElementInArray(int[] arr, int ele)* has no side effects.

Additionally you need to specify the loop needed in the algorithm. Specify a *loop_invariant* a *assignable*-clause and a *decreases*-term.

- d) Now proof your search algorithm. Should the proof fail, analyze the problem with the proof tree. Improve your specification and code and restart the proof.

Evaluation

Nearly done! Now you just need to document your experience in our questionnaire: <https://bit.ly/2F27qOS>

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

* Erforderlich

Vorwissen

1. Vorwissen *

Markieren Sie nur ein Oval.

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
- Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY

Aufgabe 1: Wiederholung Projektmanagement

2. Gab es beim Erstellen von Dateien und Ordnern Probleme? *

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

3. Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

4. Gab es beim Öffnen von Projekten und Dateien Probleme? *

Markieren Sie nur ein Oval.

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

5. Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Markieren Sie nur ein Oval.

- Ja
- Nein

6. Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Markieren Sie nur ein Oval.

- Ja
- Nein

7. Anmerkung:

KeY

8. Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

Markieren Sie nur ein Oval.

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

9. Anmerkungen

Kollaboratives Editieren

10. Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

Markieren Sie nur ein Oval.

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

11. Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Markieren Sie nur ein Oval.

- Ja
- Nein

12. Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

13. Anmerkung:

14. Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Markieren Sie nur ein Oval.

- Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.
- Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

15. Anmerkung:

16. Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

Markieren Sie nur ein Oval.

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

17. Anmerkung:

Aufgabe 3: Suche im Array

18. Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

Markieren Sie nur ein Oval.

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

19. Anmerkungen:

20. Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

Markieren Sie nur ein Oval.

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

21. Anmerkung:

22. Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

23.

Bereitgestellt von



Google Forms

Evaluation of the second user study

KollaborierbaRs

Almost there. Now all you have to do is record your experiences in this questionnaire.
We ask you to answer all questions honestly.

* Erforderlich

Prior knowledge

1. Prior knowledge *

Markieren Sie nur ein Oval.

- I am a student and I am attending the lecture "Formale Methoden im Software Entwurf"/"Formal Methods in Software Engineering" or have done so in the past.
- I am an employee at TU Darmstadt and I am working with KeY or on improving KeY.

Task 1: Editing Projects

2. Did you encounter difficulties while creating files or folders? *

Markieren Sie nur ein Oval.

- I had no problems creating files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily create files and folders.
- I was able to create files and folders, but only with the help of a supervisor.
- I was not able to create files or folders.

3. Did you encounter difficulties while deleting projects, files or folders? *

Markieren Sie nur ein Oval.

- I had no problems deleting files, folders, or projects.
- I had to get familiar with the interface briefly, but then I was able to easily delete projects, files or folders.
- I could delete projects, folders, and files only with the help of a supervisor.
- I was not able to delete files or folders.

4. Did you encounter difficulties while opening projects, files or folders? *

Markieren Sie nur ein Oval.

- I had no problems opening projects, files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily open projects, files or folders.
- I could open projects, folders, or files only with the help of a supervisor.
- I was not able to open files or folders.

5. Did the new sorting of the project structure prove to be helpful? *

Markieren Sie nur ein Oval.

- Yes
- No

6. Did the reordered context menu prove helpful?

Markieren Sie nur ein Oval.

- Yes
- No

7. Remarks

KeY

8. Was it difficult to evaluate whether a proof was successful? *

Markieren Sie nur ein Oval.

- KollaborierbaRs UI made it clear to me, whether a proof was successful or not.
- I could tell if a proof was successful only after consulting a supervisor.
- I was not able to tell, whether a proof completed successfully or not.

9. Remarks

Collaborative Editing

10. Did you notice inconsistent states when editing files collaboratively? *

Markieren Sie nur ein Oval.

- I did not notice any inconsistencies while editing.
- From time to time I noticed an inconsistency. However I was able to correct it by reloading the file.
- Collaborative editing did not work for me.

11. Has the new completion function proven to be helpful?

Markieren Sie nur ein Oval.

- Yes
- No

12. Remarks

Task 2: Incorrect specification

13. Did the proof tree help you to find the error? *

Markieren Sie nur ein Oval.

- Yes, the proof tree helped me to find the error in the specification.
- After a short explanation of the proof tree by a supervisor, it helped me to find the error.
- No, the proof tree didn't help me to find the bug in the specification.

14. Remarks

15. Did the display of the sequent help you to find the error?

Markieren Sie nur ein Oval.

- Yes, displaying the sequent helped to find the error in the specification.
- After a short explanation of the displayed sequent it helped me to find the error in the specification.
- No, displaying the sequent did not help me to find the error in the specification.

Task 3: Searching an array

16. Did the proof tree and the display of the sequent help you to verify your algorithm? *

Markieren Sie nur ein Oval.

- Yes, with the help of the proof tree and the sequent I was able to improve my specification bit by bit and finally prove it.
- With the help of the supervisor, I was able to verify my specification through the proof tree and the sequents.
- The proof tree and the sequence could not help me to specify and verify my algorithm.

17. Remarks

18. Did the display of the KeY errors within the console help you while editing the specification? *

Markieren Sie nur ein Oval.

- Yes, the KeY errors displayed in the console helped me to write the specification.
- No, the KeY errors displayed in the console didn't help me write the specification.
- I didn't notice the console.

19. Remarks:

20. Do you have any general comments on the workflow with KollaborierbaR regarding the proof tree and the display of the sequents?

Additional remarks

Auswertung der 2. Nutzerstudie - KollaborierbaR



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Inhaltsverzeichnis

1 Festgestellte Mängel während 1. Studie und Folgen	2
1.1 Projektverwaltung	2
1.2 KeY Integration	2
1.3 Kollaboratives Editieren	3
1.4 Nicht umgesetztes Feedback	3
2 Multiple-Choice Antworten	4
2.1 Vorwissen	4
2.2 Projektverwaltung	5
2.3 KeY	6
2.4 Kollaboratives Editieren	7
3 Nutzeranmerkungen aus Freitextfeldern und Betreuernotizen	8
3.1 Zusammenfassung der Betreuernotizen und dem Freitextfeedback	8
4 Neu aufgedeckte Probleme	10
5 Fazit	11
5.1 Projektverwaltung	11
5.2 KeY Integration	11
5.3 Kollaboratives Editieren	11

1 Festgestellte Mängel während 1. Studie und Folgen

Diese Sektion entspricht einem kurzen Rückblick auf die ersten Studie und stellt teilweise eine Wiederholung von deren Fazit dar. Wir geben nochmals einen kurzen Überblick der Ergebnisse für die 3 Themengebiete „Projektverwaltung“, „KeY Integration“ und „Kollaboratives Editieren“ aus der ersten Studie und gehen dann auf die Maßnahmen ein, mit welchen wir auf diese reagiert haben.

1.1 Projektverwaltung

Ergebnisse 1. Studie

Hier wurde mittels der Multiple-Choice Antworten die höchste Zufriedenheit im Vergleich zu den anderen Themengebieten festgestellt. Die schlechtesten Bewertungen erhielt die Funktionalität zum Erstellen von Dateien und Ordnern, allerdings haben auch hier nur 5% der Teilnehmer angegeben, dass sie Betreuerhilfe benötigt haben, und 35% mussten sich lediglich kurz orientieren. Inhaltlich wurden mittels der Freitextfelder und im direkten Gespräch mit dem Betreuer vor allem folgende Punkte häufig angemerkt:

Nr.	Anmerkung	Anzahl Vorkommnisse
1	Die visuelle Einrückung von geöffneten Dateien verwirrt	9
4	Versucht Datei durch Ziehen zu verschieben	8
5	Bitte Sortierung (alphabetisch und nach Ordnern/Dateien) der Sidebar	6
6	Doppelklick für Projekte öffnen	5
7	Download von Dateien	5
9	Bitte (alphabetische) Sortierung der Projektauswahl	5
10	Kontextmenü der Sidebar: Die Reihenfolge der Optionen („delete“ vor „create“) sei verwirrend, lieber an bestehenden IDEs orientieren	4
11	Einfacher Klick statt Doppelklick zum Öffnen von Dateien	4
13	Prompts sollen denjenigen Button hervorheben, welche beim Bestätigen mit Enter automatisch ausgelöst werden	4

Die häufigsten Verbesserungsvorschläge waren also an das Themengebiet „Projektmanagement“ gerichtet, obwohl es im Rahmen der Multiple-Choice Fragen am besten abgeschnitten hatte.

Maßnahmen

Bei der Umsetzung von Maßnahmen haben wir in Absprache mit dem Auftraggeber versucht, eine Balance zwischen Komplexität, Dringlichkeit und neuen Features zu finden. Die Anmerkungen 1, 5, 6, 9, 10 haben wir in Userstories gefasst und bearbeitet:

- Anpassung der Sortierung der Operationen in der Menüführung (ASOM1402) als Verbesserung für Anmerkung Nr. 10
- Sortierung von Projekten (SvP2502) als Verbesserung für Anmerkung Nr. 9
- Einheitliche Einrückung der Dateistruktur (EedD1402) als Verbesserung für Anmerkung Nr. 1
- Doppelklick auf Projektnamen öffnet Projekte (DaP1402) als Verbesserung für Anmerkung Nr. 6
- Sortierung der Sidebar (SsS2502) als Verbesserung für Anmerkung Nr. 5

1.2 KeY Integration

Ergebnisse 1. Studie

Hier meldeten 52% der Teilnehmer, dass eine kurze Orientierung notwendig gewesen sei. 14.29% benötigten Hilfe durch den Betreuer. Angemerkt durch die Teilnehmer wurden unter anderem folgende Aspekte:

Nr.	Anmerkung	Anzahl Vorkommnisse
1	Keine Orientierung	14.29%

2	Button für Einzelbeweise nicht oder nur schwer gefunden.	9
3	Rechtsklick auf Methoden zum Ausführen von Beweisen	8
22	„Open Goals“ sollten sich scrollen lassen, oder in der Zeile umbrechen	3
30	Wunsch: Mehr Details zum (Miss-)Erfolg eines Beweises	2
32	Nach Öffnen eines anderen Projekts stehen bei Open Goals noch immer die Goals vom letzten Projekt	2
41	Manuelles Speichern vor dem Beweisen stört	1

Maßnahmen

Die Anmerkungen 3, 22, 41 wurden von uns in Userstories gefasst und bearbeitet:

- Rechtsklick auf Methoden zum Beweisen von Proof Obligation (RaMzBvP2502) als Verbesserung für Anmerkung Nr. 3
- Scrollen innerhalb der Anzeige von Open Goals (SidAvOG2502) als Verbesserung für Anmerkung Nr. 22
- Automatisches Speichern vor dem Starten von Beweisen (AsvdSvB2502) als Verbesserung für Anmerkung Nr. 41

1.3 Kollaboratives Editieren

45% der Teilnehmer stellten geringfügige Probleme beim kollaborativen Arbeiten fest, z. B. inkonsistente Texte zwischen den verschiedenen Bearbeitern einer Datei. 5% empfanden das Highlighting der Änderungen anderer Teilnehmer als störend. Beispielsweise folgende Mängel wurden durch die Teilnehmer kommentiert:

Nr.	Anmerkung	Anzahl Vorkommnisse
8	Strg + Z wie gewohnt, soll nicht Änderungen anderer Personen betreffen	5
12	Wunsch: Der Cursor der anderen Mitbearbeiter soll angezeigt werden	4
14	Farben der Personen sind verwirrend (Farbe je nach geöffneter Datei), lieber eine Farbe für jede Person im ganzen Projekt	4
15	Shortcuts für Vervollständigung von Code gewünscht	3
18	Das Highlighting ist zu unauffällig, man bemerkt es zunächst nicht	3
21	Scrollen beim Schreiben von Code	3
27	Wunsch: Automatische Codevervollständigung	2
29	Wunsch: Man sollte sehen können, was andere Personen markiert haben	2

Maßnahmen

Die Anmerkungen 8, 21, 27 wurden von uns in Userstories gefasst und bearbeitet:

- Rückgängig durch Steuerung Z (RdSZ1402) als Verbesserung für Anmerkung Nr. 8
- Automatisches Scrollen beim Schreiben von Code (AsbSvC2502) als Verbesserung für Anmerkung Nr. 21
- Autovervollständigung beim Editieren (AbE1402) als Verbesserung für Anmerkung Nr. 27

1.4 Nicht umgesetztes Feedback

Im Feedback der ersten Nutzerstudie gibt es verschiedene Aspekte, welche von uns nicht verbessert wurden. Dies hat verschiedene Gründe:

Ein Grund besteht darin, dass die Umsetzung des in der Nutzerstudie erhaltenen Feedbacks parallel zur Implementierung neuer Features geschehen musste. Dies hatte zur Folge, dass nur ausgewählte, wichtige Verbesserungsvorschläge umgesetzt wurden. Des Weiteren gab es beim Feedback Aspekte, welche in der Gesamtheit aller Verbesserungsvorschläge nur selten aufkamen und die mit den Vorstellungen von uns (dem Entwicklerteam) auseinander gingen. Ein Beispiel dafür ist, dass neben der Verifikation des Programms es auch möglich sein soll, das Programm auszuführen. Dieses Feedback haben wir mit dem Auftraggeber besprochen und dann nicht umgesetzt.

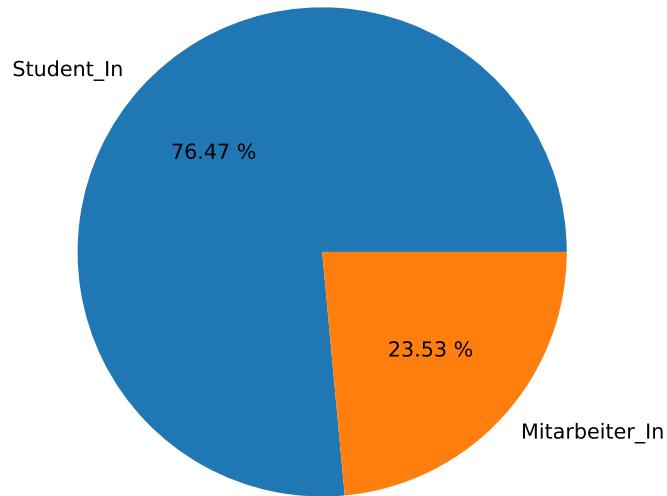
2 Multiple-Choice Antworten

Wie auch in der ersten Studie fassen wir hier zunächst zusammen welchen allgemeinen Eindruck die Probanden von den einzelnen Bestandteilen der UI aufnahmen.

Es wurden teilweise Fragen entfernt, die sich auf Themen bezogen, bei denen keine Änderungen an KollaborierbaR seit der letzten Studie erfolgt sind, und dafür neue Fragen aufgenommen, die sich direkt auf neu implementierte Features (Beweisbäume und Sequenten), oder Verbesserungen von Mängeln beziehen, die in der ersten Studie festgestellt wurden (z. B. Fragen zur Konsole oder der Autovervollständigung).

2.1 Vorwissen

Anteile der Zielgruppen an den Probanden



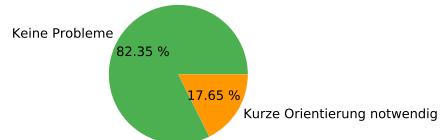
An der zweiten Studie haben 17 Personen teilgenommen, davon 13 Studierende und 4 Mitarbeiter_Innen.

Es handelt sich also um 3 weniger Studierende und um 1 weniger Mitarbeiter als bei der ersten Durchführung der Studie.

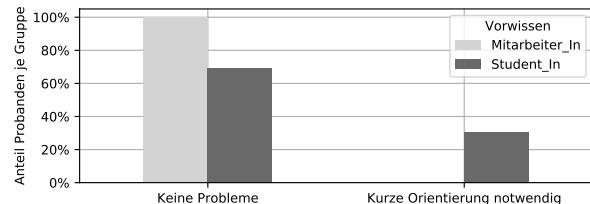
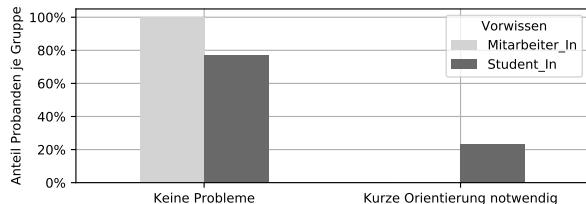
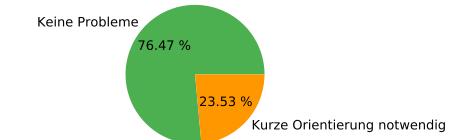
Zwar sollte man also bei der Interpretation der Ergebnisse diese Verluste bedenken, allerdings hat sich das Verhältnis zwischen den Anteilen der Mitarbeitern und Studierenden kaum verändert.

2.2 Projektverwaltung

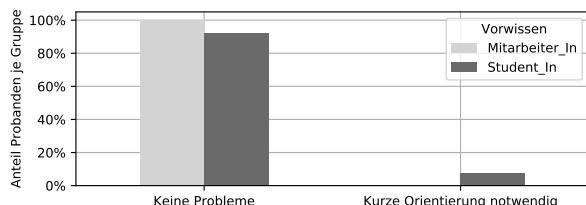
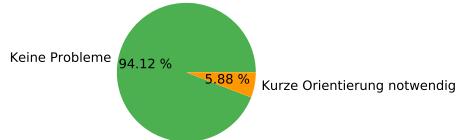
Gab es beim Erstellen von Dateien und Ordnern Probleme?



Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

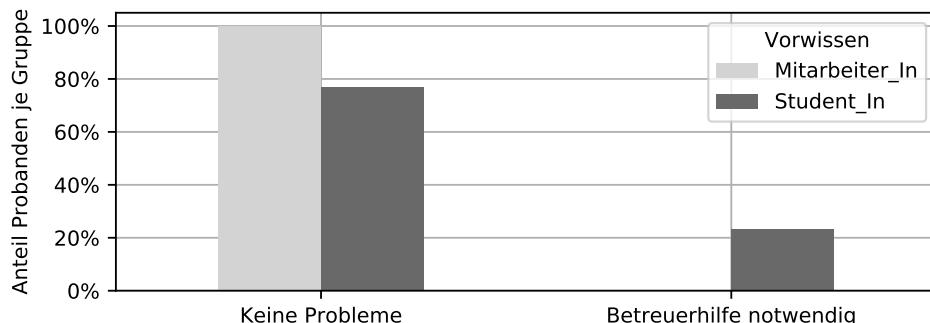
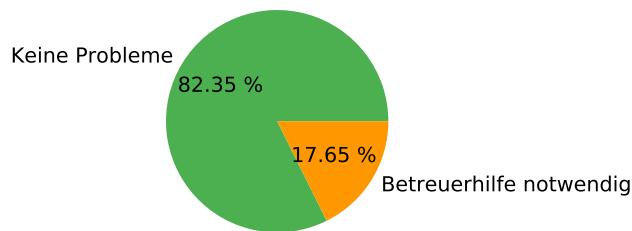


Gab es beim Öffnen von Projekten und Dateien Probleme?

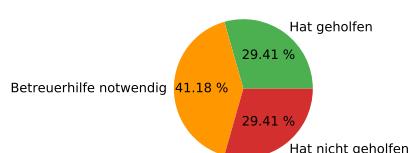


2.3 KeY

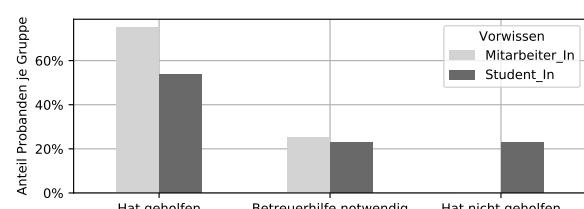
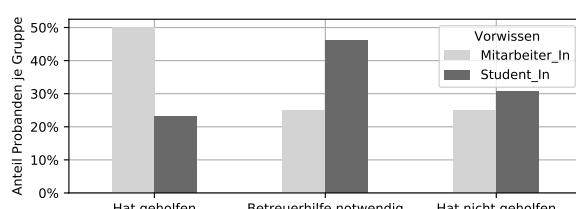
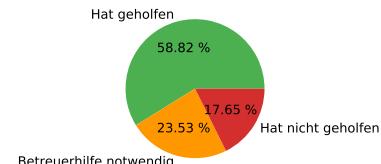
Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?



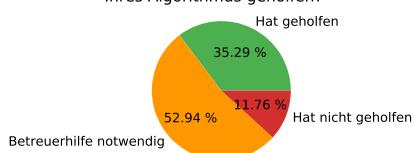
Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen?



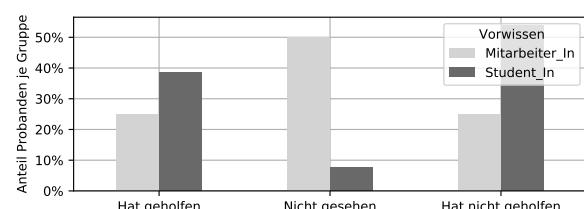
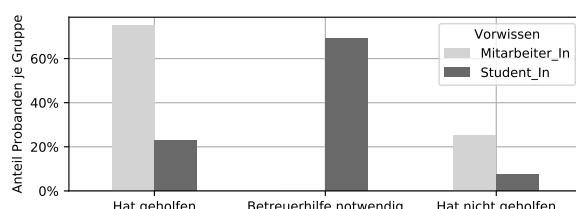
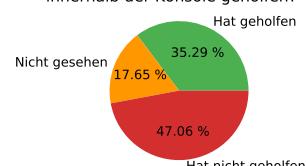
Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen?



Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

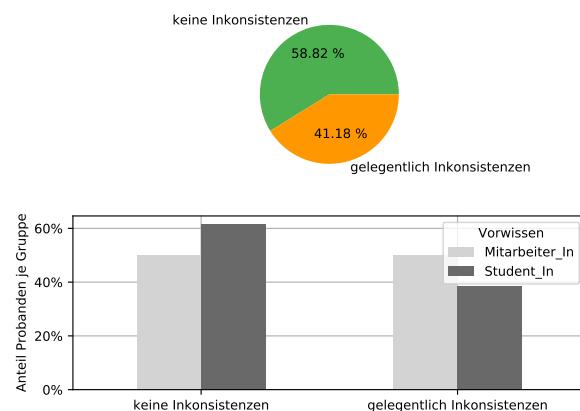


Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen?

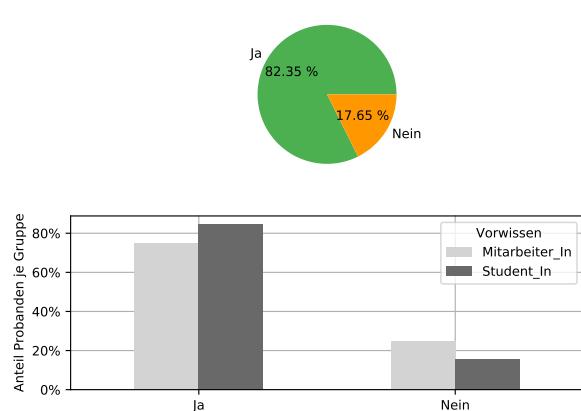


2.4 Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?



Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?



3 Nutzeranmerkungen aus Freitextfeldern und Betreuernotizen

Analog zur ersten Studie haben die Daten der Multiple-Choice Fragen vor allem ein allgemeines Stimmungsbild erfasst und erste Eindrücke geschaffen, wie gut die verschiedenen Zielgruppen (Mitarbeiter / Studierende) jeweils mit der Benutzeroberfläche zurechtgekommen sind. Das nun folgende direkte Feedback dient also wieder zur Erfassung von Verbesserungsvorschlägen und konkreten Mängeln.

3.1 Zusammenfassung der Betreuernotizen und dem Freitextfeedback

Die folgende Zusammenfassung listet alle durch die Probanden genannten Anmerkungen. Wurde eine Thematik mehrfach genannt, so wurde das Feedback zusammengefasst und notiert, von wie vielen Probanden eine vergleichbare Anmerkung geäußert wurde (wie bezeichnen diese Anzahl im Folgenden als *Priorität*).

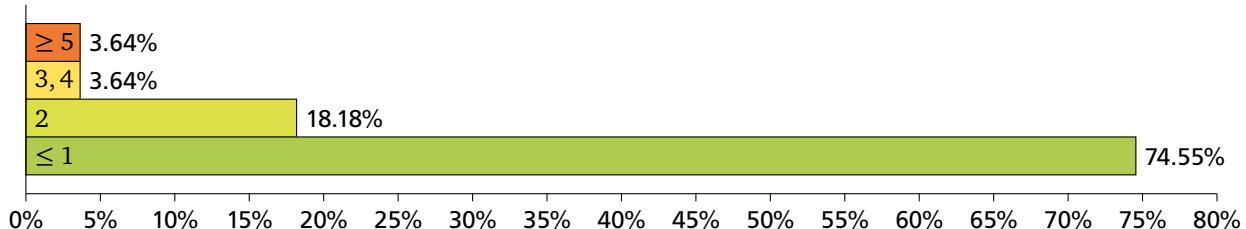
Dies soll es erlauben zu gewichten, welche Verbesserungen sich von den Benutzern besonders gewünscht werden:

Nr.	Anmerkung	Anzahl Vorkommnisse
0	JML Keywords werden durch die Auto vervollständigung nicht vorgeschlagen	6
1	Konsolen Inhalt sollte sich löschen	5
2	Mehrfaches Erstellen des selben Projekts zeigt keinen Fehler an	4
3	Einfachklick um Ordner zu öffnen, Mehrfachklick um Dateien zu öffnen ist inkonsistent	3
4	Konsoleninhalt scrollt nicht mit	2
5	Knopf zum schließen des Sequenten nicht sichtbar genug	2
6	Falls der Beweis nicht geschlossen werden konnte, soll ein rotes X im Kreis angezeigt werden.	2
7	Knopf zum Löschen des Konsoleninhalts	2
8	JML Einrückung war falsch	2
9	Name der Seite sollte nicht React App sein	2
10	Die Sortierung von Dateien und Ordnern im Projekt scheint noch nicht richtig zu funktionieren.	2
11	Enter zum Auswählen von Optionen gewünscht, zum Beispiel beim Öffnen von Dateien	2
12	Entfernen-Taste im Projektbaum	2
13	Open Goal Namen im Baum != Open Goals Tab Nummern	2
14	gefährliche Optionen der Kontextmenues (lösen) sollte rot sein	1
15	Sequent wirkt wie editierbare Datei, das verwirrt	1
16	Modals mit Escape schliessen	1
17	Delete Projekt sollte das aktuell offene Projekt löschen, statt Liste anzuzeigen	1
18	Scrollbar der Goals teilweise verdeckt	1
19	Open Goals sollte Übersichtlicher sein	1
20	Wrapping der Zeilen der Konsole	1
21	Erneutes öffnen der Datei löscht die Goals	1
22	Tastenkombination zum Starten von Beweisen/Startbutton statt Untermenüs	1
23	File Icon evtl. verwechselbar mit Ordner-Icon	1
24	Pfeiltasten vermisst, dringend	1
25	Nicht erkennbar dass sich Knoten aufklappen lassen	1
26	Create-Project: Enter = cancel	1
27	Invarianten sollte man getrennt von der Methode beweisen können	1
28	KeY Rückmeldungen/ Exceptions not very helpful	1
29	OS abhängige Einstellung für Markierungsverhalten der Dateien	1
30	KeY Exceptions parsen (Datei, Zeilenummer) evtl. im Gutter markieren	1
31	Längere Zeit bis Beweisbaum gefunden wurde	1
32	nicht bemerkt, dass man auf Beweisknoten doppelklicken kann	1
33	Musste durch Betreuer auf Lösung von Aufgabe 2 hingewiesen werden	1

34	Kontextmenü sollte sich öffnen beim Klicken in die Zeile der Datei/Ordner etc., nicht nur auf den Namen. Beim Klick auf "src" ist das etwas umständlich, genau das "src" zu treffen.	1
35	Wenn man einen sehr langen Namen für eine zu erstellende Datei eingibt und dann versucht, die Datei zu erstellen, wird man aus dem Projekt geschmissen ("No open project").	1
36	Wartehinweis wenn Proof noch läuft	1
37	Aktuell beziehen die KeY-Beweise alle Dateien des Projekts oder des Paketes mit ein. Gerade wenn mehrere Leute gleichzeitig an einem Projekt arbeiten ist es praktischer, wenn nur die geöffnete Datei genutzt wird. Sonst gibt es Fehler weil jemand in einer anderen Datei arbeitet und dieser noch nicht compilierfähig ist.	1
38	Die offenen Beweise müssen erst durch aufklappen aller Unterpunkte im Beweisbaum gesucht werden. Der Produktivität halber sollten die offenen Beweise bereits sichtbar / geöffner sein.	1
39	Gleichzeitiges Erstellen von Projekten geht nicht	1
40	Farben sind teilweise immenroch gleich	1
41	neues Projekt geöffnet alter Prooftree wird angezeigt	1
42	keine Zeilennummern inhalt der Konsole	1
43	Automatisches Speichern funktioniert nicht	1
44	Package am Anfang der Datei verhindert Beweisen	1
45	Projekte rename fehlt	1
46	Titel für die Konsole	1
47	zweimal gleicher Name vergeben	1
48	Die ganze Zeile in der Sidebar klicken (src zu kurz)	1
49	Die ersten beiden Farben sollen Blau und grün sein (Nicht rot und Orange)	1
50	Ctrl-A ist immer noch buggy!	1
51	maximale Schrittanzahl erhöhen	1
52	Goals werden doppelt angezeigt	1
53	Goals verschwinden wenn man Datei neu lädt	1
54	Im Sequenten Term markieren wo die Regel angewandt wurde	1

Insgesamt wurden also 55 Anmerkungen erfasst...

- davon 2 mit Priorität **größer gleich 5** (3.64 %)
- davon 2 mit Priorität **3 oder 4** (3.64 %)
- davon **10** mit Priorität **2** (18.18 %)
- davon **41** mit Priorität **1** (74.55 %)



4 Neu aufgedeckte Probleme

In den freien Feedback Feldern haben 7 Probanden Probleme mit der Anzeige von Fehlern in der Konsole geäußert. Somit ist die Konsole eine der größten neu aufgedeckten Schwachstellen. Aufgrund des nahenden Endes der Projektzeit, mussten wir die Wünsche der Projektleitung priorisieren. 5 Probanden haben dabei vorgeschlagen, den Konsoleninhalt zu löschen, sobald ein Fehler auftritt. 2 Probanden haben angemerkt, dass die Konsole nicht zum Anfang von neuen Fehlermeldungen scrollt. Wir sind zu dem Schluss gekommen, dass alle Probanden zufrieden sein sollten, sofern sie die letzte Fehlermeldung schnell finden können. Um die Bearbeitung der Nutzerstudie und die Wünsche des Auftraggebers zu vereinen, wurde im Rahmen der User story Ausführen von Proofscripts (AvP1103) das Scrollen der Konsole zum Anfang der letzten Konsolenmeldung implementiert, da es hier wichtig ist, die Ausgabe des Skripts in Echtzeit verfolgen zu können. Auf das restliche Feedback konnten wir im gegebenen Zeitrahmen leider nicht mehr eingehen.

5 Fazit

An dieser Stelle gehen wir noch einmal auf die drei Themengebiete „Projektverwaltung“, „KeY Integration“ und „Kollaboratives Editieren“ ein und vergleichen die Ergebnisse mit der vorangegangenen Nutzerstudie.

5.1 Projektverwaltung

Die Resonanz zur Projektverwaltung fiel schon bei der ersten Nutzerstudie im Vergleich zu den anderen Themengebieten sehr gut aus. Trotzdem konnten noch Verbesserungen erzielt werden.

In der ersten Nutzerstudie erhielt die Funktionalität zum Erstellen von Dateien und Ordnern innerhalb des Themenbereichs die schlechteste Bewertung. 5% der Teilnehmer benötigten Betreuerhilfe und 35% mussten sich kurz orientieren. Diese Werte konnten auf jeweils 0% und 17% verbessert werden.

Die Ergebnisse könnten allerdings auch dadurch beeinflusst worden sein, dass einige Teilnehmer schon in der ersten Nutzerstudie teilgenommen haben und sich diese eventuell schon an die Projektverwaltung gewöhnt haben.

5.2 KeY Integration

Die Integration von KeY schnitt im Vergleich zur ersten Nutzerstudie etwas schlechter ab. Während es in der ersten Nutzerstudie beim Interpretieren der Rückmeldung über den Erfolg eines Beweises nur in 4.8% der Fälle zu Problemen kam, brauchten in dieser Studie 17.7% der Teilnehmer Hilfe.

Wir führen die Ergebnisse darauf zurück, dass wir seit der ersten Nutzerstudie einige fortgeschrittene KeY Funktionen umgesetzt haben und diese dementsprechend in diese Nutzerstudie miteingebunden haben. Dies führte wahrscheinlich bei dem studentischen Teil unserer Teilnehmer zu erhöhter Verwirrung. Unterstützt wird diese These dadurch, dass in beiden Nutzerstudien der Anteil der wissenschaftlichen Mitarbeiter die Hilfe brauchte unter dem der Studierenden oder gar bei 0% lag. Eine Ausnahme stellte hier das Anzeigen des Sequenten dar, allerdings stellte sich im Gespräch mit den Mitarbeitern heraus, dass die Unzufriedenheit oft von den Unterschieden zur bestehenden KeY Oberfläche herrührte, insbesondere von der fehlenden Pfeiltastensteuerung.

Wir reagierten, indem wir die Pfeiltastensteuerung implementierten, siehe User Story „PiB1103“.

5.3 Kollaboratives Editieren

Das kollaborative Editieren konnte erfolgreich verbessert werden. Während in der ersten Nutzerstudie bei 45% der Teilnehmer Probleme beim gemeinsamen Bearbeiten von Dateien aufkamen, waren es in der zweiten Nutzerstudie nur 41%.

Leider handelt es sich hier nur um eine eher geringfügige Verbesserung von 4 %. Im Vergleich zur letzten Studie haben wir vor allem versucht, inkonsistente Zustände zwischen Clients zu korrigieren. Im Gespräch mit den Probanden kam allerdings auf, dass nicht unbedingt Inkonsistenzen zwischen den Editoren problematisch sind, sondern das Verhalten KollaborierbaRs bei der Herstellung von Konsistenz die Ursache des Benutzbarkeitsproblems sein könnte. Beispielsweise kann der Cursor des Benutzers das Bild verlassen, wenn ein anderer Bearbeiter Zeilen über der eigenen Position einfügt. Auch wurde geäußert, das blockweise Einfügen von Text, statt jeden Buchstaben einzeln zu behandeln, wäre eine alternative, möglicherweise bessere Eingabemethode.

Um hier weitere Verbesserungen zu erzielen, wäre es möglich, verschiedene Verhaltensarten für KollaborierbaRs Editor zu implementieren und erneut eine Studie durchzuführen, die die Zufriedenheit der Probanden mit den verschiedenen Modi misst. Der Editor könnte dann angepasst werden, dasjenige Verhalten zu implementieren, das die höchste Zufriedenheit erzielt.

Code Reviews

In dieser Sektion findet sich die für die Reviews eingesetzte Checkliste. Bei der Build-Pipeline, welche im Qualitätssicherungsdokument und in der Checkliste erwähnt wird, handelt es sich um den `make pipeline` Vorgang, der in Sektion A.5 kurz erläutert wird, und dessen Skripte und Konfigurationen sich ebenfalls dort wiederfinden.

Während der Bearbeitung des Praktikums ist es bei der Maßnahme „Code Reviews“ zu dem Problem gekommen, dass der Aufwand der Qualitätssicherungsmaßnahmen beim Verfassen des Qualitätssicherungsdokuments falsch abgeschätzt wurde. Insbesondere folgende Aspekte standen über die gesamte Implementierungszeit im Konflikt, sie in der gegebenen Zeit umzusetzen:

- die Durchführung der Nutzerstudie
- die Implementierung von Korrekturen der Mängel, die während der Studien entdeckt wurden
- die Implementierung sehr aufwendige Features in fast jedem Sprint (kollaboratives Edieren, Anbindung automatischer Beweise durch KeY, Synchronisierung von Projekt und Beweisstrukturen zwischen Clients etc.)
- das Review der implementierten Features und Korrekturen aus den Studien

Wir sahen uns daher in die Lage versetzt, diesen Konflikt nur lösen zu können durch eine Anpassung unserer Strategien aus dem Qualitätssicherungsdokument. Ziel der folgenden Anpassungen war es, den Arbeitsaufwand bewältigen zu können und dennoch die Qualitätsziele **zu erfüllen**, die durch die Reviews erreicht werden sollen (siehe Sektion 2.2.1).

1. Die durch den Auftraggeber gewünschten Features je Sprint und die Nutzerstudie als unsere Hauptmaßnahme der Qualitätssicherung sollten Vorrang erhalten.
2. Die Durchführung der Reviews wurde nachgestellt, sollte aber dennoch erfolgen.
3. Zu diesem Zweck wurde ein Entwicklungsbranch erstellt, auf welchem die Features der einzelnen Entwickler ohne Review zusammengeführt wurden.
Die Arbeit an KollaborierbaR konnte somit auf diesem Branch in der Geschwindigkeit fortgesetzt werden, die sich zur Realisierung des Praktikums als notwendig erwies.
4. Die Reviews sollten vor Ende des Projekts für jedes Feature nachgeholt werden, um das Praktikum mit dem stabilen Branch abzuschließen, wie er im Qualitätssicherungsdokument beschrieben wurde. Dieser Branch würde dann den Entwicklungsbranch ablösen.

Diese Überarbeitung aller Featureimplementierungen mit anschließendem Review ist in den letzten 2 Wochen erfolgt. Der stabile Branch wurde erzeugt.

Die ausgefüllten Checklisten dieses Korrekturvorgangs befinden sich am Ende des Anhangs in Sektion A.8.2.

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

* Erforderlich

1. Betroffener Branch *

2. Betroffene User Stories *

3. Wievielte Durchführung des Review? *

Markieren Sie nur ein Oval.

1 2 3 4



4. Reviewer *

Markieren Sie nur ein Oval.

- Marc Arnold
- Jonas Belouadi
- Anton Haubner
- David Heck
- Martin Kerscher

5. Verantwortliche(r) für Features *

Wählen Sie alle zutreffenden Antworten aus.

	Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
Entwickler	<input type="checkbox"/>				

6. Anmerkungen

Manueller Funktionstest

7. Wählen Sie alle zutreffenden Antworten aus.

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

8. Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

9. Wählen Sie alle zutreffenden Antworten aus.

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
 Code kompiliert? (`make`)
 API-Tests erfolgreich? (`make test` auf Server)

10. Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

11. Ergänzung der Spezifikation

Wählen Sie alle zutreffenden Antworten aus.

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
 Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
 Ergänzungen enthalten ein Beispiel für Rückgabewert?
 Ergänzungen enthalten Schema der beteiligten Datentypen?
 Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

12. Tests

Wählen Sie alle zutreffenden Antworten aus.

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach dem Ermessen des Reviewers ausreichend testen?
 Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

13. Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

14. Wählen Sie alle zutreffenden Antworten aus.

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

15. Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

16. Wählen Sie alle zutreffenden Antworten aus.

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

17. Wählen Sie alle zutreffenden Antworten aus.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

18. Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn *alle* Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

19. Wählen Sie alle zutreffenden Antworten aus.

- Wurde das Code Review bestanden?

20. Anmerkungen

Bereitgestellt von



Codeanhänge



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Contents

1	Codeanhänge	2
1.1	Spezifizierte Rest Schnittstellen	2
1.1.1	ProjectController	2
1.1.2	ProofController	9
1.1.3	LinterController	17
1.2	Zentrale Klassen	18
1.2.1	Sidebar.tsx	18
1.2.2	ProofSyncController.ts	26

1 Codeanhänge

In diesem Dokument werden die geforderten Codeausschnitte aufgelistet. Da unsere Qualitätssicherungsmaßnahmen die Dokumentation der Server API umfassen, folgt zunächst in Sektion 1.1 der Code des Servers, der die API bereitstellt.

In Sektion 1.2 sind dann weiterhin 10 Seiten Code zentraler Klassen zu finden.

1.1 Spezifizierte Rest Schnittstellen

Diese Sektion enthält die Implementierung der in unserer Spezifikation genannten HTTP Rest Schnittstellen.

1.1.1 ProjectController

Die *ProjectController* Klasse enthält die Implementierung aller Endpunkte, welche mit */project/*** über Http angesprochen werden.

```
1 package server;
2
3 import events.DeletedFileEvent;
4 import events.DeletedProjectEvent;
5 import events.RenamedFileEvent;
6 import events.UpdatedFileEvent;
7 import events.UpdatedProjectEvent;
8 import java.io.File;
9 import java.io.FileNotFoundException;
10 import java.io.FileOutputStream;
11 import java.io.IOException;
12 import java.io.OutputStreamWriter;
13 import java.io.Writer;
14 import java.nio.charset.StandardCharsets;
15 import java.util.LinkedList;
16 import java.util.List;
17 import java.util.NoSuchElementException;
18 import java.util.Scanner;
19 import javax.servlet.http.HttpServletRequest;
20 import org.springframework.beans.factory.annotation.Autowired;
21 import org.springframework.context.ApplicationEventPublisher;
22 import org.springframework.http.HttpStatus;
23 import org.springframework.http.ResponseEntity;
24 import org.springframework.web.bind.annotation.CrossOrigin;
25 import org.springframework.web.bind.annotation.PathVariable;
26 import org.springframework.web.bind.annotation.RequestBody;
27 import org.springframework.web.bind.annotation.RequestMapping;
28 import org.springframework.web.bind.annotation.RequestMethod;
29 import org.springframework.web.bind.annotation.RequestParam;
30 import org.springframework.web.bind.annotation.ResponseBody;
31 import org.springframework.web.bind.annotation.RestController;
32 import org.springframework.web.servlet.HandlerMapping;
33 import projectmanagement.FileItem;
34 import projectmanagement.FileUpdateData;
35 import projectmanagement.FolderItem;
36 import projectmanagement.Item;
37 import projectmanagement.OpenedFileResponse;
38
39 /**
40 * This is a rest controller for manipulating the project file structure. This includes for example
41 * deleting or adding files.
42 *
43 * @author Marc Arnold, David Heck
44 */
45 @RestController
```

```

46  @CrossOrigin
47  @RequestMapping("/projects")
48  public class ProjectController {
49      private static final String projectPath = "projects";
50
51      @Autowired private ApplicationEventPublisher applicationEventPublisher;
52
53      /**
54      * That method handels requests to /listProjects and creates a list of project names.
55      *
56      * @return a List containing Stings of the Names form of the Folders in the Projects
57      *         folder(currently hardcoded)
58      */
59      @RequestMapping(value = "", method = RequestMethod.GET)
60      public List<String> listProjects() {
61          final List<String> projects = new LinkedList<String>();
62
63          final File file = new File(projectPath);
64          final File[] files = file.listFiles();
65
66          for (final File f : files) {
67              projects.add(f.getName());
68          }
69
70          return projects;
71      }
72
73      /**
74      * That method handles requests to /{projectname} and creates a folderItem object which models the
75      * folder structure of the given folder/project name. The object will later be marshalled through
76      * Java Spring, resulting in a JSON object.
77      *
78      * @param request is given in the http request.
79      * @return the content of a chooses Projekt (currently hardcoded) in the form of a folder
80      */
81      @RequestMapping(value = "/{projectname}", method = RequestMethod.GET)
82      public FolderItem showProject(
83          @PathVariable("projectname") String projectname, HttpServletRequest request) {
84          // Get the File/Folder form the file system
85          final File file = new File(projectPath);
86          final File[] files = file.listFiles();
87
88          final File selected = selectProjectFromArray(files, projectname);
89
90          return createFolderItem(selected);
91      }
92
93      /**
94      * Helper function for recursivly creating a folderItem object from a given file structure.
95      *
96      * @param file A file from the file system
97      * @return A Folder and its content
98      */
99      public FolderItem createFolderItem(File file) {
100         final List<Item> entries = new LinkedList<Item>();
101
102         // Adding the content of the Folder to a list if it is a file just add it if its a folder add it
103         // and call the method again recursivly
104         for (final File f : file.listFiles()) {
105             if (f.isFile()) {
106                 entries.add(new FileItem(f.getName()));
107             } else if (f.isDirectory()) {
108                 entries.add(createFolderItem(f));
109             }
110         }
111         return new FolderItem(entries, file.getName());

```

```

112     }
113
114     /**
115      * Selectes a folder/project from a given file structure and returns it.
116      *
117      * @param files List of the content of a folder
118      * @param name Name of the to be selected Folder
119      * @return The folder matching the giving name or null if it does not exist
120      */
121     public File selectProjectFromArray(File[] files, String name) {
122         for (final File f : files) {
123             if (f.getName().equals(name)) {
124                 return f;
125             }
126         }
127         return null;
128     }
129
130     /**
131      * That method handels request to /** (which should represend a path to a file) and returns the
132      * contents of a file and its name.
133      *
134      * @param request to the file, which is supposed to be opened.
135      * @return object containing filename and filetext (object for marshalling)
136      */
137     @RequestMapping(value = "/**", method = RequestMethod.GET)
138     @ResponseBody
139     public ResponseEntity<?> openFile(HttpServletRequest request) throws IOException {
140
141         // Get the file path for the request resource
142         String path =
143             ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
144             .substring(1);
145
146         try {
147
148             final File file = new File(path);
149
150             Scanner scan = new Scanner(/*scanners allow to read a file until a delimiter*/ file, "utf-8");
151
152             // Check whether the requested file is empty
153             if (!scan.hasNext()) {
154                 scan.close();
155                 return new ResponseEntity<OpenedFileResponse>(
156                     new OpenedFileResponse(file.getName(), ""), HttpStatus.OK);
157             }
158
159             final String content =
160                 scan.useDelimiter("\Z").next(); // read until end of file (Z delimiter)
161             // ^ using a scanner may not be optimal (could cause overhead),
162             // but simplifies this code so much, that we keep it for now
163
164             scan.close();
165             return new ResponseEntity<OpenedFileResponse>(
166                 new OpenedFileResponse(file.getName(), content), HttpStatus.OK);
167         } catch (FileNotFoundException e) {
168             e.printStackTrace();
169             return new ResponseEntity<String>(
170                 "File could not be found. The following path was used for search:" + path,
171                 HttpStatus.NOT_FOUND);
172         } catch (NoSuchElementException e) {
173             e.printStackTrace();
174             return new ResponseEntity<String>(
175                 "Read Error. Error while reading the request file: " + path,
176                 HttpStatus.INTERNAL_SERVER_ERROR);
177         } catch (IllegalStateException e) {

```

```

178     e.printStackTrace();
179     return new ResponseEntity<String>(
180         "Read Error. Error while reading the request file: " + path,
181         HttpStatus.INTERNAL_SERVER_ERROR);
182     }
183   }
184
185 /**
186 * This Method handles the creation of Files and Folders
187 *
188 * @param type Type of the kind of structure to be created can be file or folder
189 * @param request HttpServletRequest in order to get the full path
190 * @return Returns a HttpStatus depending on whether the right type was given.
191 * @throws IOException when a new file could not be created
192 */
193 @RequestMapping(value = "/{projectname}/**", method = RequestMethod.PUT)
194 @ResponseBody
195 public ResponseEntity<?> createFile(
196     @PathVariable("projectname") String projectname,
197     @RequestParam("type") String type,
198     HttpServletRequest request)
199     throws IOException {
200
201     // Removes the first character from the path string, we need this because java.io.File need a
202     // path that does not start with a "/"
203     String path =
204         ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
205         .substring(1);
206
207     final File file = new File(path);
208
209     // Java createNewFile and mkdir are not able to create a file if a file
210     // with the same name already exists. Therefore, if someone tries to create
211     // a file with the same name, return a Http Bad Request Response
212     if (file.exists()) {
213         return new ResponseEntity<String>(
214             "There already exists a file with the same name you try to create",
215             HttpStatus.BAD_REQUEST);
216     }
217
218     // check which kind of structure should be created
219     if (type.equals("file")) {
220         file.createNewFile();
221     } else if (type.equals("folder")) {
222         file.mkdir();
223     } else {
224         // Wrong type parameter was selected, respond with Bad request code
225
226         return new ResponseEntity<>(
227             "Wrong type parameter was chosen in the request. To create a file, please select file or folder
228             as type.",
229             HttpStatus.BAD_REQUEST);
230     }
231
232     final UpdatedProjectEvent event = new UpdatedProjectEvent(this, projectname);
233     applicationEventPublisher.publishEvent(event);
234
235     // If everything was good, return the new project structure together with a HTTP OK response
236     // code
237     return new ResponseEntity<FolderItem>(showProject(projectname, request), HttpStatus.OK);
238
239 /**
240 * This Method handles the deletion of files and folders
241 *
242 * @param request HttpServletRequest in order to get the full path

```

```

243 * @return Returns a HttpStatus depending on whether the file to be deleted exists.
244 */
245 @RequestMapping(value = "/{projectname}/**", method = RequestMethod.DELETE)
246 @ResponseBody
247 public ResponseEntity<?> deleteFile(
248     @PathVariable("projectname") String projectname, HttpServletRequest request) {
249
250     // Removes the first character from the path string, we need this because java.io.File need a
251     // path that does not start with a "/"
252     final String path =
253         ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
254         .substring(1);
255     final String filePath;
256     {
257         final int separatorIdx = path.indexOf('/');
258         if (separatorIdx == -1 || separatorIdx + 1 >= path.length()) {
259             return new ResponseEntity<>(
260                 "A file path withing the project needs to be specified.", HttpStatus.NOT_FOUND);
261         }
262
263         filePath = path.substring(separatorIdx + 1);
264     }
265
266     final File file = new File(path);
267     // check if the given path actually leads to a valid directory
268     if (!file.exists()) {
269         return new ResponseEntity<>(
270             "The file you try to delete does not exist.", HttpStatus.NOT_FOUND);
271     } else {
272         try {
273
274             List<String> deletedFiles = delete(path, file);
275             final DeletedFileEvent event =
276                 new DeletedFileEvent(this, projectname, filePath, deletedFiles);
277             applicationEventPublisher.publishEvent(event);
278
279             return new ResponseEntity<FolderItem>(showProject(projectname, request), HttpStatus.OK);
280         } catch (IOException e) {
281             e.printStackTrace();
282             return new ResponseEntity<>(
283                 "File exists, but could still not be deleted", HttpStatus.INTERNAL_SERVER_ERROR);
284         }
285     }
286 }
287
288 /**
289 * This Method handles the deletion of projects
290 *
291 * @param request HttpServletRequest in order to get the full path
292 * @return Returns a HttpStatus depending on whether the file to be deleted exists.
293 */
294 @RequestMapping(value = "/{projectname}", method = RequestMethod.DELETE)
295 @ResponseBody
296 public ResponseEntity<?> deleteProject(
297     @PathVariable("projectname") String projectname, HttpServletRequest request) {
298
299     // Removes the first character from the path string, we need this because java.io.File need a
300     // path that does not start with a "/"
301     String path =
302         ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
303         .substring(1);
304
305     final File file = new File(path);
306     // check if the given path actually leads to a valid directory
307     if (!file.exists()) {
308         return new ResponseEntity<>(

```

```

309         "The file you try to delete does not exist.", HttpStatus.NOT_FOUND);
310     } else {
311         try {
312
313             List<String> deletedFiles = delete(path, file);
314             final DeletedProjectEvent event = new DeletedProjectEvent(this, projectname, deletedFiles);
315             applicationEventPublisher.publishEvent(event);
316
317             // IMPORTANT: This function exists separated from the default deleteFile function, because
318             // if we delete a project, we don't have a Json Object to return
319             return new ResponseEntity<>(HttpStatus.OK);
320         } catch (IOException e) {
321             e.printStackTrace();
322             return new ResponseEntity<>(
323                 "File exists, but could still not be deleted", HttpStatus.INTERNAL_SERVER_ERROR);
324         }
325     }
326 }
327
328 /**
329 * Recursive helper method for deleting a file or folder that is called with a base path and
330 * returns the absolute path of all deleted files or folders
331 *
332 * @param path path of the second parameter file
333 * @param file file or directory to delete
334 * @throws IOException exception if file cannot be deleted (for example it might not exist)
335 */
336 private List<String> delete(String path, File file) throws IOException {
337     List<String> result = new LinkedList<>();
338     // if the current file is not a file but a directory we need to delete its content first.
339     if (file.isDirectory()) {
340         // if the current directory is not empty list its content and call delete recursively
341         if (file.listFiles().length != 0) {
342             for (File f : file.listFiles()) {
343                 result.addAll(delete(path + "/" + f.getName(), f));
344             }
345         }
346     }
347     result.add(path);
348     // if the current directory is an empty directory or a file, delete it
349     final boolean deleted = file.delete();
350     if (!deleted) {
351         throw new IOException("Could not delete file");
352     }
353     return result;
354 }
355
356 /**
357 * This method handles updates to files and folders. For files/folders it is possible to rename
358 * the filename.
359 *
360 * <p>For files it is also possible to update the filecontent. But it is not possible to update
361 * fileName and fileContent with one methodcall.
362 *
363 * <p>Example JSON for updating fileContent: { fileContent: 'Some Content' }
364 *
365 * <p>Example JSON for updating fileName: { fileName: 'path' }
366 *
367 * <p>The incoming JSON is marshalled into a FileUpdateData object.
368 */
369 @RequestMapping(value = "/{projectname}/**", method = RequestMethod.POST)
370 @ResponseBody
371 public ResponseEntity<> updateFile(
372     @PathVariable("projectname") String projectname,
373     @RequestBody FileUpdateData updateData,
374     HttpServletRequest request) {

```

```

375
376     // Get the file path for the request resource
377     // substring(1) remove the "/..." at the beginning of a path
378     final String path =
379         ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
380             .substring(1);
381
382     final String originalPath;
383     {
384         final int separatorIdx = path.indexOf('/', path.indexOf('/') + 1);
385         if (separatorIdx == -1 || separatorIdx + 1 >= path.length()) {
386             return new ResponseEntity<>(
387                 "A file path within the project needs to be specified.", HttpStatus.NOT_FOUND);
388         }
389
390         originalPath = path.substring(separatorIdx + 1);
391     }
392
393     // Rename a file or a folder
394     if (updateData.fileContent == null) {
395
396         final File file = new File(path);
397         // substring(1) remove the "/..." at the beginning of a path
398         final String path2 = updateData.fileName.substring(1);
399         final boolean success = file.renameTo(new File(path2));
400
401         if (success) {
402             final String newPath;
403             {
404                 final int separatorIdx = path2.indexOf('/', path2.indexOf('/') + 1);
405                 if (separatorIdx == -1 || separatorIdx + 1 >= path2.length()) {
406                     return new ResponseEntity<>(
407                         "A file path within the project needs to be specified as target path.",
408                         HttpStatus.NOT_FOUND);
409                 }
410
411                 newPath = path2.substring(separatorIdx + 1);
412             }
413
414             final RenamedFileEvent event =
415                 new RenamedFileEvent(this, projectname, originalPath, newPath);
416             applicationEventPublisher.publishEvent(event);
417
418             return new ResponseEntity<>(showProject(projectname, request), HttpStatus.OK);
419         } else {
420             return new ResponseEntity<>("The file could no be renamed.", HttpStatus.BAD_REQUEST);
421         }
422     } else {
423         // If you are in this branch, the fileContent wasn't null. That implies that you do have a
424         // file
425         // and that the
426         // caller of that functions wants to update the content, not the name of the file.
427         try {
428             // Oldversion: Had problems with encoding
429             // final BufferedWriter writer = new BufferedWriter(new FileWriter(path));
430             final File f = new File(path);
431             final Writer writer =
432                 new OutputStreamWriter(new FileOutputStream(f), StandardCharsets.UTF_8);
433             writer.write(updateData.fileContent);
434             writer.close();
435
436             final UpdatedFileEvent event = new UpdatedFileEvent(this, projectname, originalPath);
437             applicationEventPublisher.publishEvent(event);
438
439             return new ResponseEntity<>(showProject(projectname, request), HttpStatus.OK);
440         } catch (IOException e) {

```

```

441         e.printStackTrace();
442         return new ResponseEntity<>(
443             "Something went wrong while updating the file content.", HttpStatus.BAD_REQUEST);
444     }
445   }
446 }
447 }
```

1.1.2 ProofController

Die *ProofController* Klasse enthält die Implementierung aller Endpunkte, welche mit */proof/*** über Http angesprochen werden.

```

1 package server;
2
3 import events.ConsoleMessageEvent;
4 import events.ErrorEvent;
5 import events.UpdatedProofEvent;
6 import events.UpdatedProofHistoryEvent;
7 import java.util.HashMap;
8 import java.util.LinkedList;
9 import java.util.List;
10 import java.util.Observer;
11 import java.util.Optional;
12 import java.util.Set;
13 import java.util.concurrent.ConcurrentHashMap;
14 import java.util.regex.Matcher;
15 import java.util.regex.Pattern;
16 import javax.servlet.http.HttpServletRequest;
17 import org.springframework.beans.factory.annotation.Autowired;
18 import org.springframework.context.ApplicationEventPublisher;
19 import org.springframework.http.HttpStatus;
20 import org.springframework.http.ResponseEntity;
21 import org.springframework.web.bind.annotation.CrossOrigin;
22 import org.springframework.web.bind.annotation.PathVariable;
23 import org.springframework.web.bind.annotation.RequestBody;
24 import org.springframework.web.bind.annotation.RequestMapping;
25 import org.springframework.web.bind.annotation.RequestMethod;
26 import org.springframework.web.bind.annotation.RequestParam;
27 import org.springframework.web.bind.annotation.ResponseBody;
28 import org.springframework.web.bind.annotation.RestController;
29 import org.springframework.web.servlet.HandlerMapping;
30 import proofutil.KeYWrapper;
31 import proofutil.ObligationResult;
32 import proofutil.ProofResult;
33 import repository.File;
34 import repository.MethodContract;
35 import repository.ObligationService;
36
37 /**
38 * Controller which provides access to KollaborierbaRs KeY functionalities. Those are:
39 *
40 * <p>- Running proofs on a file stored on the server - Temporarily storing proofs, such that other
41 * clients working on the same file can access them - Permanently storing a history of proof results
42 * for a proof obligation for each file This is utilized by the client, such that users can retrieve
43 * and review proofs at a later time.
44 *
45 * @author Jonas Belouadi
46 * @author Anton Haubner {@literal <anton.haubner@outlook.de>}
47 */
```

```

48  @RestController
49  @CrossOrigin
50  @RequestMapping("/proof")
51  public class ProofController {
52      // ProjectFilePath -> (ObligationId -> ObligationResult)
53      private ConcurrentHashMap<String, HashMap<Integer, List<ObligationResult>>> obligationResults =
54          new ConcurrentHashMap<>();
55
56      /**
57      * Required, to send events between Spring controllers.
58      *
59      * <p>In this class, it is used to inform {@link synchronization.ProofSyncController} about
60      * changes to the proof result history and temporarily stored proof results.
61      */
62      @Autowired private ApplicationEventPublisher applicationEventPublisher;
63
64      @Autowired private ObligationService obligationService;
65
66      @Autowired private FileService fileService;
67
68      /**
69      * Try to prove all proof obligations in a .java file or by index if an index is provided
70      *
71      * @param className class name of the file in which the proofs shall be run
72      * @param obligationIdxs the indices of the obligations to prove. Counted from top to bottom in
73      * the corresponding Java source file
74      * @param macro the path to the macro file to use for the proof, if present
75      * @return the proof results
76      */
77      @RequestMapping(value = "/{className}.java", method = RequestMethod.GET)
78      @ResponseBody
79      public ResponseEntity<ProofResult> runProof(
80          @PathVariable final String className,
81          @RequestParam("obligationIdxs") final Optional<List<Integer>> obligationIdxs,
82          @RequestParam("macro") final Optional<String> macro,
83          final HttpServletRequest request) {
84          // Get the file path for the request resource
85          final PathData pathData = decodePath(request);
86          final String projectFilePath = pathData.projectFilePath;
87
88          // Listens for text that should be sent to the console.
89          // Emits an event that triggers the broadcast of the text to all users
90          // that are connected to the file being proven
91          final Observer console =
92              (observable, object) -> {
93                  String message = (String) object;
94                  ConsoleMessageEvent event =
95                      new ConsoleMessageEvent(this, pathData.projectName, pathData.filePath, message);
96                  applicationEventPublisher.publishEvent(event);
97              };
98
99          // Listens for error texts that should be displayed in an error notification
100         // Emits an event that triggers the broadcast of the error to all users
101         // that are connected to the file being proven
102         final Observer errorObserver =
103             (observable, object) -> {
104                 String message = (String) object;
105                 ErrorEvent event = new ErrorEvent(this, pathData.projectName, pathData.filePath, message);
106                 applicationEventPublisher.publishEvent(event);
107             };
108
109         final KeyWrapper key = new KeyWrapper(projectFilePath, console, errorObserver);
110
111         // KeyWrapper provides KeY functionalities to this API controller
112         Optional<String> macroContentsOptional = Optional.empty();
113

```

```

114     if (macro.isPresent()) {
115         // Read the macro file
116         String macroContents = fileService.getCurrent(pathData.projectName + macro.get());
117         if (macroContents != "") {
118             System.out.println("ProofController: Using macro:\n" + macro.get());
119             macroContentsOptional = Optional.of(macroContents);
120         }
121     }
122
123     // prove by index if index is present. ternary operator can be replaced with ifPresentOrElse if
124     // Java 9 is used or higher
125     final ProofResult result =
126         obligationIdxs.isPresent()
127             ? key.proveContractByIdxs(className, obligationIdxs.get(), macroContentsOptional)
128             : key.proveAllContracts(className, macroContentsOptional);
129
130     key.dispose();
131     return new ResponseEntity<ProofResult>(result, HttpStatus.OK);
132 }
133
134 /**
135 * Lists all obligation indices for the given file, for which there are saved proofs available on
136 * the server.
137 *
138 * <p>The returned indices can be used to retrieve saved proofs using for example {@link
139 * #getCurrentProof} or {@link getHistoricProof}.
140 *
141 * @param className Name of the class, for which we want to list obligation indices of saved
142 *      proofs.
143 * @param request Object describing the HTTP request, which triggered this method. Filled in by
144 *      Spring.
145 * @return indices of saved proof results, counted per file from top to bottom.
146 */
147 @RequestMapping(value = "/**/{className}.java/obligation", method = RequestMethod.GET)
148 public Set<Integer> listSavedObligations(
149     @PathVariable final String className, final HttpServletRequest request) {
150
151     final PathData pathData = decodePath(request);
152     final String projectFilePath = pathData.projectFilePath;
153
154     final File file = obligationService.getFile(projectFilePath);
155     return file.getObligations().keySet();
156 }
157
158 /**
159 * Temporarily store the latest prove result for a proof. It is lost as soon as the server is shut
160 * down.
161 *
162 * <p>This feature is used by KollaborierbaR to always share the latest proof result by any
163 * developer with everyone working on the same file. Using this route, such a temporarily result
164 * can be uploaded.
165 *
166 * <p>This method will inform {@link synchronization.ProofSyncController} about the change.
167 *
168 * @param className Name of the class, for which we want to save a temporary proof result.
169 * @param obligationIdx Index of the proof obligation to which the uploaded result belongs,
170 *      counted from top to bottom in the given Java file.
171 * @param request Object describing the HTTP request, which triggered this method. Filled in by
172 *      Spring.
173 * @param obligationResult Proof result to be stored.
174 */
175 @RequestMapping(
176     value = "/**/{className}.java/obligation/{obligationIdx}/last",
177     method = RequestMethod.POST)
178 public void uploadCurrentObligationResult(
179     @PathVariable final String className,

```

```

180     @PathVariable final int obligationIdx,
181     final HttpServletRequest request,
182     @RequestBody final ObligationResult obligationResult) {
183
184     final PathData pathData = decodePath(request);
185     final String filePath = pathData.projectFilePath;
186
187     System.out.println(
188         "ProofController: Got obligation result for path "
189         + filePath
190         + ": "
191         + obligationResult.getResultMsg());
192
193     File file = obligationService.getFile(filePath);
194     MethodContract methodContract = obligationService.getMethodContract(file, obligationIdx);
195     System.out.println("Target name: " + obligationResult.getTargetName());
196
197     ObligationResult savedObligationResult = obligationService.save(obligationResult);
198     methodContract.setLast(savedObligationResult);
199     obligationService.save(methodContract);
200
201     final UpdatedProofEvent event =
202         new UpdatedProofEvent(this, pathData.projectName, pathData.filePath, pathData.obligationId);
203     applicationEventPublisher.publishEvent(event);
204 }
205
206 /**
207 * Retrieves the temporary proof result stored by {@link #uploadCurrentObligationResult}.
208 *
209 * @param className Name of the class, for which we want to retrieve a temporary proof result.
210 * @param obligationIdx Index of the proof obligation for which the latest temporary results shall
211 * be retrieved, counted from top to bottom in the given Java file.
212 * @param request Object describing the HTTP request, which triggered this method. Filled in by
213 * Spring.
214 * @return Temporarily stored proof result, or a NOT_FOUND (404) response, if there is no such
215 * result stored.
216 */
217 @RequestMapping(
218     value = "/**/{className}.java/obligation/{obligationIdx}/last",
219     method = RequestMethod.GET)
220 public ResponseEntity<ObligationResult> getCurrentProof(
221     @PathVariable final String className,
222     @PathVariable final int obligationIdx,
223     final HttpServletRequest request) {
224
225     final PathData pathData = decodePath(request);
226     final String filePath = pathData.projectFilePath;
227
228     File file = obligationService.getFile(filePath);
229     if (file.getObligations().containsKey(obligationIdx)) {
230         return new ResponseEntity<>(
231             file.getObligations().get(obligationIdx).getLast(), HttpStatus.OK);
232     }
233
234     return new ResponseEntity<>(HttpStatus.NOT_FOUND);
235 }
236
237 /**
238 * Retrieves a proof result from the history. See class description for more information on the
239 * proof history feature.
240 *
241 * @param className Name of the class, for which we want to retrieve a proof result from the
242 * history.
243 * @param obligationIdx Index of the proof obligation for which the result shall be retrieved,
244 * counted from top to bottom in the given Java file.
245 * @param historyIdx unique history id of the saved proof. See {@link #getHistoryItems} for more

```

```

246     *      information about history ids.
247     * @param request Object describing the HTTP request, which triggered this method. Filled in by
248     *      Spring.
249     * @return Saved proof result, or a NOT_FOUND (404) response, if there is no such result stored.
250     */
251     @RequestMapping(
252         value = "/**/{className}.java/obligation/{obligationIdx}/history/{historyIdx}",
253         method = RequestMethod.GET)
254     public ResponseEntity<ObligationResult> getHistoricProof(
255         @PathVariable final String className,
256         @PathVariable final int obligationIdx,
257         @PathVariable final int historyIdx,
258         final HttpServletRequest request) {
259
260         Optional<ObligationResult> requested = obligationService.findObligationResultById(historyIdx);
261         if (requested.isPresent()) {
262             return new ResponseEntity<>(requested.get(), HttpStatus.OK);
263         }
264
265         return new ResponseEntity<>(HttpStatus.BAD_REQUEST);
266     }
267
268     /**
269     * Retrieves a list of ids of all proof results stored in the history. See class description for
270     * more information on the proof history feature.
271     *
272     * <p>These ids can be used to retrieve a saved proof result using {@link #getHistoricProof}.
273     *
274     * @param className Name of the class, for which we want to retrieve the list of ids
275     * @param obligationIdx Index of the proof obligation for which the history ids shall be
276     *      retrieved.
277     * @param request Object describing the HTTP request, which triggered this method. Filled in by
278     *      Spring.
279     * @return List of ids of saved proof results. The history id of a saved result is an indicator of
280     *      its age. Bigger ids mean the saved result is more recent. So sorting them by descending
281     *      order sorts them in order of their age, beginning with the newest saved proofs.
282     */
283     @RequestMapping(
284         value = "/**/{className}.java/obligation/{obligationIdx}/history",
285         method = RequestMethod.GET)
286     public List<Long> getHistoryItems(
287         @PathVariable final String className,
288         @PathVariable final int obligationIdx,
289         final HttpServletRequest request) {
290
291         final PathData pathData = decodePath(request);
292         final String projectFilePath = pathData.projectFilePath;
293
294         final List<Long> ids = new LinkedList<>();
295
296         final File file = obligationService.getFile(projectFilePath);
297         if (file.getObligations().containsKey(obligationIdx)) {
298             List<ObligationResult> history = file.getObligations().get(obligationIdx).getHistory();
299             for (ObligationResult result : history) {
300                 ids.add(result.getId());
301             }
302         }
303
304         return ids;
305     }
306
307     /**
308     * Adds a proof result to the history of saved results. See class description for more information
309     * on the proof history feature.
310     *
311     * <p>This method will inform {@link synchronization.ProofSyncController} about the changed

```

```

312     * history.
313     *
314     * @param className Name of the class, for which we want to save a result
315     * @param obligationIdx Index of the proof obligation for which a proof result shall be saved.
316     * @param request Object describing the HTTP request, which triggered this method. Filled in by
317     *      Spring.
318     * @return history id of the saved proof, see {@link #getHistoryItems} for more information on
319     *      these ids.
320     */
321     @RequestMapping(
322         value = "/**/{className}.java/obligation/{obligationIdx}/history",
323         method = RequestMethod.POST)
324     @ResponseBody
325     public ResponseEntity<Long> addToHistory(
326         @PathVariable final String className,
327         @PathVariable final int obligationIdx,
328         @RequestBody final ObligationResult obligationResult,
329         final HttpServletRequest request) {
330
331         final PathData pathData = decodePath(request);
332         final String projectFilePath = pathData.projectFilePath;
333
334         System.out.println("ProofController: About to save a new obligation result to history");
335
336         final File file = obligationService.getFile(pathData.projectFilePath);
337         final MethodContract methodContract = obligationService.getMethodContract(file, obligationIdx);
338
339         ObligationResult savedObligationResult = obligationService.save(obligationResult);
340         methodContract.addToHistory(savedObligationResult);
341         obligationService.save(methodContract);
342         obligationService.save(savedObligationResult);
343
344         final UpdatedProofHistoryEvent event =
345             new UpdatedProofHistoryEvent(
346                 this, pathData.projectName, pathData.filePath, pathData.obligationId);
347         System.out.println(
348             "ProofController: Publishing updated history. There are now "
349             + methodContract.getHistory().size()
350             + " results stored in the history of obligation "
351             + obligationIdx);
352
353         applicationEventPublisher.publishEvent(event);
354
355         return new ResponseEntity(savedObligationResult.getId(), HttpStatus.OK);
356     }
357
358     /**
359     * Deletes a proof result from the history of saved results. See class description for more
360     * information on the proof history feature.
361     *
362     * <p>This method will inform {@link synchronization.ProofSyncController} about the change.
363     *
364     * @param className Name of the class, for which we want to delete a result
365     * @param obligationIdx Index of the proof obligation for which a proof result shall be deleted.
366     * @param historyIdx unique history id of the result which shall be deleted. See {@link
367     *      #getHistoryItems} for more information about history ids.
368     * @param request Object describing the HTTP request, which triggered this method. Filled in by
369     *      Spring.
370     * @return history id of the result to be deleted, see {@link #getHistoryItems} for more
371     *      information on these ids. If there is no result matching the parameters, NOT_FOUND (404)
372     *      will be returned.
373     */
374     @RequestMapping(
375         value = "/**/{className}.java/obligation/{obligationIdx}/history/{historyIdx}",
376         method = RequestMethod.DELETE)
377     public ResponseEntity<Long> deleteFromHistory(

```

```

378     @PathVariable final String className,
379     @PathVariable final int obligationIdx,
380     @PathVariable final int historyIdx,
381     final HttpServletRequest request) {
382
383     final PathData pathData = decodePath(request);
384     final String projectFilePath = pathData.projectFilePath;
385
386     System.out.println(
387         "ProofController: About to delete obligation result for "
388         + projectFilePath
389         + " on obligation id "
390         + obligationIdx
391         + " from history");
392
393     final Optional<ObligationResult> toDeleteOptional =
394         obligationService.findObligationResultById(historyIdx);
395     if (!toDeleteOptional.isPresent()) {
396         return new ResponseEntity(HttpStatus.NOT_FOUND);
397     } else {
398         ObligationResult toDelete = toDeleteOptional.get();
399         toDelete.setMethodContract(null);
400         toDelete = obligationService.save(toDelete);
401
402         final UpdatedProofHistoryEvent event =
403             new UpdatedProofHistoryEvent(
404                 this, pathData.projectName, pathData.filePath, pathData.obligationId);
405         System.out.println(
406             "ProofController: Publishing updated history after removal of item " + historyIdx + ".");
407
408         applicationEventPublisher.publishEvent(event);
409
410         return new ResponseEntity(historyIdx, HttpStatus.OK);
411     }
412 }
413
414 /**
415 * Since Spring does not decode arbitrary paths ('**') into method parameters for us, we have to
416 * extract them from the request object.
417 *
418 * <p>This method just provides a basic extraction of the whole path without any further
419 * processing. See {@link #decodePath} for a similar method, which provides more information on
420 * the contents of the path.
421 *
422 * @param request Spring generated object, describing a request. Can be obtained by simply adding
423 * the type to the parameter list of a method with a {@link RequestMapping}.
424 * @return The full path used in {@code request}, with the leading '/proof/' part removed.
425 */
426 private static String extractPath(final HttpServletRequest request) {
427     return ((String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE))
428         .substring(7); // remove '/proof/' from the start of the path.
429 }
430
431 /**
432 * Simple data structure utilized by {@link #decodePath} to provide information extracted from a
433 * request path.
434 */
435 private static class PathData {
436     /**
437      * Index of a proof obligation, as counted from top to bottom in its Java source file.
438      *
439      * <p>It may be set to {@code -1}, if there was no obligation index specified within the path.
440      */
441     public final int obligationId;
442     /** Name of the project referenced in the request path. */
443     public final String projectName;

```

```

444     /** Path to a file within the project */
445     public final String filePath;
446     /** {@link projectName} and {@link filePath} concatenated with an '/' */
447     public final String projectFilePath;
448
449     public PathData(
450         final int obligationId,
451         final String projectName,
452         final String filePath,
453         final String projectFilePath) {
454         this.obligationId = obligationId;
455         this.projectName = projectName;
456         this.filePath = filePath;
457         this.projectFilePath = projectFilePath;
458     }
459 }
460
461 /**
462 * Since Spring does not decode arbitrary paths ('/*') into method parameters for us, we have to
463 * extract them from the request object.
464 *
465 * <p>It also extracts additional information from the path. See {@link PathData} for the
466 * extracted information.
467 *
468 * @param request Spring generated object, describing a request. Can be obtained by simply adding
469 * the type to the parameter list of a method with a {@link RequestMapping}.
470 * @return The full path used in {@code request}, with the leading '/proof/' part removed.
471 */
472 private static PathData decodePath(final HttpServletRequest request) {
473     // regex, which is used to extract information from the request path by
474     // defining groups
475     final String regex =
476
477         "\\\\[\\/]proof\\\\/(?<ProjectName>[^\\/]+)\\\\/(?<Path>[^\\/.]+\\\\.java)(\\\\obligation(\\\\/(?<ObligationId>\\d+)\\/.+)?";
478
479     // Retrieving the request path from the request object
480     final String input =
481         (String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE);
482
483     // Applying the regex
484     final Pattern p = Pattern.compile(regex);
485     final Matcher m = p.matcher(input);
486     m.matches();
487
488     // Extracting information
489     final String projectName = m.group("ProjectName");
490     final String path = m.group("Path");
491
492     // If there was no obligation id specified in the path, set it to -1
493     int obligationId = -1;
494     try {
495         obligationId = Integer.parseInt(m.group("ObligationId"));
496     } catch (final Exception e) {
497         obligationId = -1;
498         System.out.println(
499             "ProofController: Could not identify obligation id. Might have been omitted to access
500             ..../obligation and is no error in that case.");
501     }
502
503     final String projectFilePath = projectName + "/" + path;
504
505     return new PathData(obligationId, projectName, path, projectFilePath);
506 }
507 }
```

1.1.3 LinterController

Die *LinterController* Klasse enthält die Implementierung unserer */lint* Http Route.

```
1 package server;
2
3 import java.util.ArrayList;
4 import java.util.Arrays;
5 import java.util.List;
6 import linter.Diagnostic;
7 import linter.JavaJdtLinter;
8 import linter.JavaSourceMemoryObject;
9 import org.springframework.web.bind.annotation.CrossOrigin;
10 import org.springframework.web.bind.annotation.RequestBody;
11 import org.springframework.web.bind.annotation.RequestMapping;
12 import org.springframework.web.bind.annotation.RequestMethod;
13 import org.springframework.web.bind.annotation.RequestParam;
14 import org.springframework.web.bind.annotation.ResponseBody;
15 import org.springframework.web.bind.annotation.RestController;
16
17 /** Naive HTTP API (RESTful?) for linting (java) source code */
18 @RestController
19 public class LinterController {
20     private final JavaJdtLinter linter = new JavaJdtLinter();
21
22     /**
23      * implements "/lint" routing
24      *
25      * @param name class name of java source file
26      * @param source source code of file Example: POST request to http://myserver/lint?name=MyClass
27      *           with the source code of MyClass.java within the request body.
28      */
29     @RequestMapping(value = "/lint", method = RequestMethod.POST)
30     @ResponseBody
31     @CrossOrigin
32     public List<Diagnostic> lint(
33         @RequestParam("name") String filename, @RequestBody(required = false) String source) {
34         // check, whether there is a body
35         if (source != null) {
36             // Cut away .java of the file name for the java compiler
37             final String classname = cutFileExtension(filename);
38
39             try {
40                 return linter.check(NSArray.asList(new JavaSourceMemoryObject(classname, source)));
41             } catch (final Exception e) {
42                 e.printStackTrace();
43
44                 // TODO proper error handling
45                 return new ArrayList<>();
46             }
47         } else {
48             return new ArrayList<>();
49         }
50     }
51
52     /**
53      * Method cuts of the .java file extension of a string, if .java is at the end of the string
54      *
55      * @param filename of the file, for which the extension should be cut off
56      * @return name without the file extension
57      * @example cutFileExtension("Main.java") == "Main" cutFileExtension("LoremIpsum") == "LoremIpsum"
58      */
59     private String cutFileExtension(final String filename) {
60         String classname = filename;
61     }
}
```

```

62     // Length of the name is necessary
63     final int length = filename.length();
64
65     // If length is too small, it is not possible that there is an .java at the end
66     if (length < 5) {
67         classname = filename;
68     } else {
69         // String that might contain the file extension
70         final String fileextension = filename.substring(length - 5);
71
72         // If last 5 chars of string match .java, cut off the last 5 chars
73         if (fileextension.equals(".java")) {
74             classname = filename.substring(0, length - 5);
75         }
76     }
77
78     return classname;
79 }
80 }
```

1.2 Zentrale Klassen

Da in Sektion 1.1 bereits ein Teil des Servercodes gezeigt wurde, haben wir uns entschieden, hier Klassen des Client aufzulisten.

Sektion 1.2.1 zeigt hierbei eine Komponente der Nutzeroberfläche und Sektion 1.2.2 eine Klasse, die Logik zur Synchronisation von Clients bereitstellt.

1.2.1 Sidebar.tsx

Das folgende Code Listing enthält eine unserer zentralen Klassen des Clients: Die Sidebar. Diese wird neben dem Editor angezeigt.

```

1 import React, { RefObject } from 'react';
2
3 import '@fortawesome/fontawesome-free/css/all.min.css';
4 import FontAwesome from 'react-fontawesome';
5
6 import ProjectTreeView from './ProjectTreeView';
7 import OpenGoalsView from './OpenGoalsView';
8 import ProofTabView from './ProofTabView';
9
10 import { Nav, NavItem, NavLink, TabContent, TabPane } from 'reactstrap';
11 import classnames from 'classnames';
12
13 import './sidebar.css';
14 import Project from '../../Project';
15 import ProofsState from '../../key/ProofsState';
16 import ObligationResult from '../../key/netdata/ObligationResult';
17
18 /**
19 * Allows to display content in a window left from the main content
20 * (usually an editor).
21 *
22 * It's length is variable and can be changed by the user.
23 * It can also be hidden and restored by clicking a button.
24 *
25 * Currently it contains a project tree view, a list of open goals and proof inspection few.
26 * @see {@link ProjectTreeView}
```

```

27   * @see {@link OpenGoalsView}
28   * @see {@link ProofTabView}
29   */
30 export default class Sidebar extends React.Component<Props, State> {
31   private sideBar: RefObject<HTMLDivElement>;
32
33   // Minimum width of this sidebar.
34   // This setting determines, how much the user can control the sidebar size.
35   private static readonly minWidth: number = 220;
36
37   constructor(props: Props) {
38     super(props);
39
40     // handles to rendered sub components
41     this.sideBar = React.createRef();
42
43     // determine, whether the initial project property is set and
44     // contains anything. If not, the sidebar will be collapsed initially.
45     const isProjectValid =
46       this.props.project && Object.keys(this.props.project).length !== 0; // the project property is set
47       ↪ to something
48     // ^ the project object must not be empty
49
50     this.enableTab = this.enableTab.bind(this);
51     this.setUrgentTab = this.setUrgentTab.bind(this);
52     this.resetUrgentTab = this.resetUrgentTab.bind(this);
53     this.toggle = this.toggle.bind(this);
54     this.moveSplitBar = this.moveSplitBar.bind(this);
55
56     this.state = {
57       // current width of the sidebar
58       sidebarWidth: 220,
59       // whether the sidebar is currently hidden, or not
60       collapsed: isProjectValid
61         ? // only display the sidebar initially, if a project is set
62           false
63         : true,
64       // currently open tab, see NavLink tags within the render method
65       activeTab: '1',
66       // initially no tab should be marked as urgent
67       urgentTab: '-1',
68     };
69   }
70
71   /**
72    * Invert visibility of the sidebar by collapsing it, or
73    * restoring it.
74    */
75   private toggle() {
76     this.setState({
77       collapsed: !this.state.collapsed,
78     });
79   }
80
81   /**
82    * The sidebar is built from different tabs with different content.
83    * Using this function, the current tab can be changed.
84    *
85    * @param tab id of the tab that shall be shown. Use the same Id as in the NavLink definitions within
86    → the render method.
87    */
88   private enableTab(tab: string): void {
89     if (this.state.activeTab !== tab) {
90       this.setState({
91         activeTab: tab,
92       });

```

```

91     }
92   }
93
94 /**
95 * The sidebar is built from different tabs with different content.
96 * Using this function, one tab can be set as urgent with a color that indicates urgency.
97 *
98 * @param tab id of the tab that should be set urgent. Use the same Id as in the NavLink definitions
99 *           within the render method.
100 */
101 private setUrgentTab(tab: string): void {
102   if (this.state.urgentTab !== tab && this.state.activeTab !== tab) {
103     this.setState({
104       urgentTab: tab,
105     });
106   }
107
108 /**
109 * The sidebar is built from different tabs with different content.
110 * Using this function, the urgent tab can be reset to a normal tab
111 * param tab - the id of the tab or undefined
112 */
113 private resetUrgentTab(tab?: string | undefined): void {
114   if (!tab || this.state.urgentTab === tab) {
115     this.setState({
116       urgentTab: '-1',
117     });
118   }
119 }
120
121 /**
122 * Handles resizing the sidebar when user holds the left mouse button
123 * down on the bar, that splits sidebar and main content.
124 *
125 * This method is only meant to be called as an event handler for mouse movement.
126 *
127 * @param e mouse event, for example generated by onMouseMove
128 */
129 private moveSplitBar(e: React.MouseEvent<HTMLDivElement>) {
130   // whatever usually happens, when pressing the mouse down on the split
131   // bar (bar that splits view into sidebar and main content), suppress
132   // it.
133   e.preventDefault();
134
135   // this function will translate mouse movement into adaptions to
136   // the sidebar's width.
137   const movementHandler = (movementEvent: MouseEvent) => {
138     // same effect as above
139     movementEvent.preventDefault();
140
141     if (this.sidebar.current != null) {
142       // we want to the splitbar to be wherever the user is currently
143       // moving the mouse.
144       //
145       // Therefore, the new width of the sidebar is calculated,
146       // by subtracting the sidebar's current offset to the left window
147       // border from the x coordinate of the mouse.
148       const newWidth =
149         movementEvent.pageX -
150         this.sidebar.current.getBoundingClientRect().left;
151
152       // check, that the new width does not violate the minimum and
153       // maximum width restrictions. Also check, that the mouse is still
154       // within the window.
155       if (

```

```

156         newWidth > Sidebar.minWidth &&
157         movementEvent.pageX < window.innerWidth
158     ) {
159         this.setState({
160             sidebarWidth: newWidth,
161         });
162         // Make other components recalculate their width (Im looking at you ace)
163         window.dispatchEvent(new Event('resize'));
164     }
165 }
166 };
167
168 // the user pressed down on the split bar.
169 // Until they let go, the above handler shall process mouse movement
170 document.addEventListener('mousemove', movementHandler);
171
172 // remove the installed handlers, as soon as the user lets go of
173 // the split bar (stops pressing the left button)
174 const mouseupHandler = (mouseUpEvent: MouseEvent) => {
175     document.removeEventListener('mousemove', movementHandler);
176     document.removeEventListener('mouseup', mouseupHandler);
177 };
178
179 document.addEventListener('mouseup', mouseupHandler);
180 }
181
182 /**
183 * Observes changes in the sidebars properties and reacts to some of them:
184 *
185 * - If the currently open project changes, the sidebar opens itself and the
186 *   project view
187 * - If the currently open proof changes and not all goals are closed, the
188 *   proof view is shown
189 *
190 * Also, a resize event is dispatched every time properties changed, since
191 * it fixes visual bugs of the Ace editor, which is displayed right from the
192 * sidebar.
193 *
194 * @param prevProps properties state before they got updated
195 */
196 public componentDidUpdate(prevProps: Props) {
197     // has the project been changed?
198     if (prevProps.project !== this.props.project) {
199         // if so, the sidebar should definitely be visible
200         this.setState({
201             collapsed: false,
202         });
203         this.enableTab('1');
204     }
205
206     // otherwise, if any proof or proof history changed, open the proof view tab
207     // (given, that there are any available proofs)
208     else if (
209         prevProps.proofsState !== this.props.proofsState &&
210         this.props.proofsState.numOfObligationsWithAvailableProofs() > 0
211     ) {
212         this.setUrgentTab('3');
213     }
214
215     if (
216         prevProps.openedPath.length !== this.props.openedPath.length ||
217         prevProps.openedPath.some(
218             (value, index) => value !== this.props.openedPath[index]
219         )
220     ) {
221         this.resetUrgentTab();

```

```

222     }
223
224     // This fixes the bug, where the height of the ace isn't correct until you resize the sidebar
225     window.dispatchEvent(new Event('resize'));
226   }
227
228 /**
229 * React lifecycle rendering method.
230 * {@link https://reactjs.org/docs/react-component.html#render}
231 *
232 * See constructor documentation for information on what is being rendered.
233 */
234 public render() {
235   // Visibility of split bar components will be controlled by
236   // employing the css display property.
237   // This function helps to calculate appropriate values for it.
238   const genVisibilityString = (collapsed: boolean) =>
239     collapsed ? 'none' : '';
240
241   // css manipulation of sidebar components, depending on the
242   // current width and visibility
243   const sidebarStyleMod = {
244     width: this.state.sidebarWidth,
245     display: genVisibilityString(this.state.collapsed),
246   };
247
248   const restoreHandleStyleMod = {
249     display: genVisibilityString(!this.state.collapsed),
250   };
251
252   // We want to display a dropdown, where any obligation can be selected, for
253   // which proofs exist.
254   // For this, we need to convert all available proofs into a format, which
255   // can be used with the dropdown display library (react-select)
256   const obligations = this.props.proofsState
257     .getAllRecentObligationResults()
258     .map(obligationResult => {
259       return {
260         value: obligationResult.obligationIdx,
261         label: `${obligationResult.obligationIdx}: ${{
262           obligationResult.targetName
263         }}`,
264       };
265     })
266     .sort((lhs, rhs) => lhs.value - rhs.value);
267   // ^we want to show the obligation selection in the same order, as
268   // obligations appear within their java file
269
270   return (
271     <>
272       {/* this bar will be used to restore the sidebar,
273          if it has been collapsed. */}
274       <div
275         className="sidebarRestoreHandle"
276         style={restoreHandleStyleMod}
277         onClick={this.toggle}
278       >
279         <FontAwesome className="sidebarRestoreButton" name="angle-right" />
280       </div>
281
282       <div className="sidebar" ref={this.sidebar} style={sidebarStyleMod}>
283         {/* this bar will be used to resize the sidebar */}
284         <div className="sidebar-split-bar" onMouseDown={this.moveSplitBar} />
285
286         {/* pressing this button shall collapse the sidebar */}

```

```

287     <div className="sidebarToggleButton" onClick={this.toggle}>
288         <FontAwesome name="chevron-circle-left" />
289     </div>
290
291     /* tab view */
292     <div className="sidebarContent">
293         <Nav tabs>
294             <NavItem>
295                 <NavLink
296                     className={classnames({
297                         active: this.state.activeTab === '1',
298                         urgent: this.state.urgentTab === '1',
299                     })}
300                     onClick={() => {
301                         this.resetUrgentTab('1');
302                         this.enableTab('1');
303                     }}
304                 >
305                     Project
306                     </NavLink>
307                 </NavItem>
308             <NavItem>
309                 <NavLink
310                     className={classnames({
311                         active: this.state.activeTab === '2',
312                         urgent: this.state.urgentTab === '2',
313                     })}
314                     onClick={() => {
315                         this.resetUrgentTab('2');
316                         this.enableTab('2');
317                     }}
318                 >
319                     Open Goals
320                     </NavLink>
321                 </NavItem>
322             <NavItem>
323                 <NavLink
324                     className={classnames({
325                         active: this.state.activeTab === '3',
326                         urgent: this.state.urgentTab === '3',
327                     })}
328                     onClick={() => {
329                         this.resetUrgentTab('3');
330                         this.enableTab('3');
331                     }}
332                 >
333                     Proof
334                     </NavLink>
335                 </NavItem>
336             </Nav>
337         <div className="tabWrapper">
338             <div className="tabContents">
339                 <TabContent activeTab={this.state.activeTab}>
340                     <TabPane tabId="1">
341                         <div id="projectTree">
342                             <ProjectTreeView
343                                 onOpenFile={this.props.onOpenFile}
344                                 onDeleteFile={this.propsonDeleteFile}
345                                 onCreateFile={this.props.onCreateFile}
346                                 onDeleteProject={this.props.onDeleteProject}
347                                 onUpdateFileName={this.props.onUpdateFileName}
348                                 project={this.props.project}
349                                 openedPath={this.props.openedPath}
350                         />

```

```

351         </div>
352     </TabPane>
353     <TabPane tabId="2">
354       <OpenGoalsView
355         proofsState={this.props.proofsState}
356         displayFormula={this.props.displayFormula}
357       />
358     </TabPane>
359     <TabPane tabId="3">
360       <ProofTabView
361         obligationOptions={obligations}
362         proofsState={this.props.proofsState}
363         obligationIdOfLastUpdatedProof={
364           this.props.obligationIdOfLastUpdatedProof
365         }
366         displaySequent={this.props.displayFormula}
367         saveObligationResult={this.props.saveObligationResult}
368         deleteObligationResult={this.props.deleteObligationResult}
369       />
370     </TabPane>
371   </TabContent>
372   </div>
373   </div>
374   </div>
375   </div>
376   </>
377 );
378 }
379 }

380 interface Props {
381   /** Currently open project structure, as rendered in the first sidebar tab */
382   project: Project;
383   /** Currently available proofs as accessible in the third sidebar tab */
384   proofsState: ProofsState;
385   /** Needed by {@link ProofTabView} to show the last updated proof */
386   obligationIdOfLastUpdatedProof: number;
387   /** Called, when a file is double clicked in the project tree view */
388   onOpenFile: (path: string[]) => void;
389   /** Called, when the delete context option is selected on an item in the project tree view */
390   onDeleteFile: (path: string[]) => void;
391   /**
392    * Called, when the user selects the file creation option in the project tree view.
393    *
394    * @param path file path to where the file shall be created. The items in the array
395    * are folders except for the last item, which shall be the name of the
396    * new file
397    * @param type decides, whether a file or a folder is being created.
398    * Possible values: 'file' or 'folder'
399    */
400   onCreateFile: (path: string[], type: string) => void;
401   /** Called, when the project deletion context option is selected in the project tree view */
402   onDeleteProject: (path: string) => void;
403   /** Called, when a file is being renamed by the user using the project tree view */
404   onUpdateFileName: (path: string[]) => void;
405   /** Path of currently opened file. Used by the {@link ProjectTreeView} to highlight it */
406   openedPath: string[];
407   /** Called, whenever a proof node or open goal is selected, to display its sequent formula. */
408   displayFormula: (formula: string) => void;
409   /** Called, whenever the user wants to save a proof to the history */
410   saveObligationResult: (obligationResult: ObligationResult) => void;
411   /** Called, whenever the user wants to erase a proof from the history */
412   deleteObligationResult: (obligationIdx: number, historyIdx: number) => void;
413 }
414 }
```

```
415
416 interface State {
417   /** current width in pixels of the sidebar */
418   sidebarWidth: number;
419   /** whether the sidebar is currently hidden, or not */
420   collapsed: boolean;
421   /** id of the currently opened tab, the the NavLink items generated within the {@link Sidebar.render}
422   → method */
422   activeTab: string;
423   /** id of a tab where something urgent is going on */
424   urgentTab: string;
425 }
```

1.2.2 ProofSyncController.ts

Diese Klasse empfängt mittels einer Websocketverbindung, die durch `StompService.ts` bereitgestellt wird, Events vom Server, die den Client über Änderungen an Beweisen informieren, die über Methoden der aktuell offenen Datei geführt wurden. Dies ist notwendig, da diese Änderungen beispielsweise durch andere Clients, erfolgt sein könnten, die an der gleichen Datei arbeiten. Auf diese Weise wird der gleiche Zustand (bezüglich Daten die Beweise betreffen) zwischen allen Clients hergestellt.

```
1 import { StompService } from '../StompService';
2 import KeYApi from '../key/KeYApi';
3
4 import ProofNode from '../key/prooftree/ProofNode';
5 import ObligationResult from '../key/netdata/ObligationResult';
6
7 import { serverAddress } from '../constants';
8
9 /**
10 * Manages synchronization of proofs between clients working on the same file.
11 *
12 * For this, it subscribes to a
13 * <a href="https://stomp.github.io/stomp-specification-1.2.html#SUBSCRIBE">STOMP destination</a>
14 * at the backend server via a websocket.
15 * This destination corresponds to the currently opened file and using the
16 * destination, the backend sends us event messages, which inform about changes, like new proofs run by
17 * other clients.
18 *
19 * For more information, consult the backend documentation of class
20 * synchronization.ProofSyncController.
21 */
22
23 export default class ProofSyncController {
24     private stompService: StompService;
25     private observer: ProofEventObserver;
26     private currentProjectName?: string;
27     private currentFilePath?: string[];
28     private currentTopic?: string;
29
30     /**
31      * @param stompService - access to a websocket connection with the server, needed for synchronization
32      * between clients.
33      * @param observer - this observer will be informed about changes to proofs, this controller witnesses.
34      * Usually it is used to update the UI when a change happens.
35      */
36     constructor(stompService: StompService, observer: ProofEventObserver) {
37         this.stompService = stompService;
38         this.observer = observer;
39
40         this.setObligationResult = this.setObligationResult.bind(this);
41         this.genTopic = this.genTopic.bind(this);
42         this.openFile = this.openFile.bind(this);
43         this.closeFile = this.closeFile.bind(this);
44     }
45
46     /**
47      * Helper method for generating the correct STOMP destination/topic we need
48      * to subscribe to, if we want the server to inform us about changes regarding
49      * a specific file.
50      */
51     private genTopic(projectName: string, path: string[]): string {
52         return `/user/projects/proof/${projectName}/${path.join('/')}`;
53     }
54
55     /**
56      * Whenever a file is opened, the application should call this function.
57      * It causes this controller to subscribe to the corresponding STOMP destination, see class description.
58      */
```

```

56     * After calling this function, the observer will be informed about updates to proofs regarding this
57     * file.
58     */
59     public openFile(projectName: string, path: string[]): Promise<void> {
60       const topic = this.genTopic(projectName, path);
61
62       let maybeUnsubscribe: Promise<void>;
63       if (this.currentTopic != null) {
64         maybeUnsubscribe = this.closeFile();
65       } else {
66         maybeUnsubscribe = Promise.resolve();
67       }
68
69       return maybeUnsubscribe.then(() =>
70         this.stompService
71           .safeSubscribe(
72             topic,
73             msg => {
74               try {
75                 const event: ProofEvent = JSON.parse(msg.body);
76
77                 console.log(`incoming proof event`, event);
78
79                 switch (event.eventType) {
80                   case ProofEventType.UpdatedProof:
81                     this.observer.onUpdatedProof(event);
82                     break;
83
84                   case ProofEventType.UpdatedProofHistory:
85                     this.observer.onUpdatedHistory(event);
86                     break;
87                 }
88               } catch (e) {
89                 console.error('Failed to parse server event');
90                 console.error(e);
91               }
92             },
93             {}
94           )
95           .then(() => {
96             this.currentProjectName = projectName;
97             this.currentFilePath = path;
98             this.currentTopic = topic;
99           })
100      );
101    }
102
103 /**
104  * Whenever a file is closed, the application should call this function.
105  * It causes this controller to unsubscribe from the corresponding STOMP destination, see class
106  * description.
107  */
108 public closeFile(): Promise<void> {
109   if (this.currentTopic == null) {
110     return Promise.reject(
111       'There is no topic set, we can not close the current proof context'
112     );
113   } else {
114     return this.stompService.safeUnsubscribe(this.currentTopic).then(() => {
115       this.currentTopic = undefined;
116       this.currentFilePath = undefined;
117       this.currentProjectName = undefined;
118     });
119   }
}

```

```

120
121  /**
122   * Sets the most recent proof result for the currently opened file / obligation at the
123   * server, so that other clients may access it.
124   */
125  public setObligationResult(obligationResult: ObligationResult): void {
126    if (
127      this.currentTopic == null ||
128      this.currentFilePath == null ||
129      this.currentProjectName == null
130    ) {
131      console.error(
132        'There is no topic set, we can not determine the current proof context'
133      );
134    } else {
135      KeYApi.uploadLatestProof(
136        this.currentProjectName,
137        this.currentFilePath.join('/'),
138        obligationResult
139      );
140    }
141  }
142}
143
144 export enum ProofEventType {
145   /** someone editing the file ran a proof and changed the most recent proof result */
146   UpdatedProof = 'UpdatedProofEvent',
147   /** someone editing the file saved or deleted a proof result from the proof result history */
148   UpdatedProofHistory = 'UpdatedProofHistoryEvent',
149 }
150
151 /**
152  * The server informs the client about changes to proofs of the currently opened
153  * file by sending {@link ProofSyncController} events, which indicate, what changed.
154  *
155  * Those events conform to this interface.
156  */
157 export interface ProofEvent {
158   eventType: ProofEventType;
159   projectName: string;
160   filePath: string;
161   obligationIdx: number;
162 }
163
164 export interface UpdatedProofEvent extends ProofEvent {}
165 export interface UpdatedProofHistoryEvent extends ProofEvent {}
166
167 /**
168  * Enables user of {@link ProofSyncController} to listen for changes to the
169  * proofs regarding the currently opened project.
170  * See constructor documentation of {@link ProofSyncController}.
171  */
172 interface ProofEventObserver {
173   onUpdatedProof(event: ProofEvent): void;
174
175   onUpdatedHistory(event: ProofEvent): void;
176 }

```

Konfigurationen der Buildtools



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Contents

1 Konfigurationen der Buildtools	2
1.1 Makefile für das gesamte Projekt	2
1.2 Tools für den Client	3
1.2.1 Makefile für Client	3
1.2.2 NPM Config	4
1.2.3 TSLint	5
1.2.4 ESLint	6
1.2.5 Prettier	7
1.3 Tools für den Server	7
1.3.1 Makefile für Server	7
1.3.2 Gradle Config für Server (build.gradle)	8
1.3.3 Checkstyle	11
1.3.4 Spotbugs	15
1.4 Beispieldurchführung	16

1 Konfigurationen der Buildtools

1.1 Makefile für das gesamte Projekt

Das ganze Projekt kann mittels eines Makefiles gebaut und ausgeführt werden. Dieses vereint alle Buildtools und Stylechecker.

Durch das Makefile werden also alle folgenden Schritte des Buildprozesses zur Verfügung gestellt, die von den konkreten Tools abstrahieren:

Kommando	Effekt
'make setup'	Download aller Abhängigkeiten, die zum Bauen von Kollaborierbar benötigt werden (mittels npm/gradle wrapper)
'make check'	Ausführung der Stylechecker und Tools für statische Analyse (Prettier, Checkstyle, Spotless / ESLint, TSLint, PMD, Spotbugs)
'make'	Bauen des Clients und Servers
'make test'	Ausführung der Server API Tests
'make pipeline'	Ausführung aller obigen Schritte
'make deploy'	Bau eines tar.gz Archivs, welches das Deployment von KollaborierbaR ermöglicht
'make format'	Ausführung automatischer Code Formatter (Prettier, Spotless)
'make clean'	Entfernung aller Produkte des Buildprozesses

Für die Code Reviews wurde beispielsweise `make pipeline` eingesetzt.

Das Makefile delegiert die genannten Aufgaben an zwei weitere Makefiles, jeweils für den Client und den Server, die weiter unten zu finden sind.

```
1 CLIENT_DIR = client
2 SERVER_DIR = server
3
4 # indicate, that submodules are not files
5 .PHONY: setup check format client linter test clean deploy
6
7 all: server client
8
9 # install dependencies etc.
10 # (will download submodules, if we are not running within a Jenkins instance)
11 setup:
12     if [ -z "${RUNNING_IN_JENKINS}" ]; then \
13         git submodule update --init --recursive; \
14     fi;
15     $(MAKE) -C $(CLIENT_DIR) setup
16     $(MAKE) -C $(SERVER_DIR) setup
17
18 # run static analysis tools
19 check:
20     $(MAKE) -C $(CLIENT_DIR) check
21     $(MAKE) -C $(SERVER_DIR) check
22
23 # Run automatic source code formatters
24 format:
25     $(MAKE) -C $(CLIENT_DIR) format
26     $(MAKE) -C $(SERVER_DIR) format
27
28 # build
29 client:
30     $(MAKE) -C $(CLIENT_DIR)
31
32 server:
33     $(MAKE) -C $(SERVER_DIR)
```

```

34
35 # run unit tests
36 test:
37     $(MAKE) -C $(CLIENT_DIR) test
38     $(MAKE) -C $(SERVER_DIR) test
39
40 # run a complete ci pipeline, like the Jenkinsfile would
41 # (this needs to be updated, when the Jenkinsfile is updated)
42 pipeline: setup check all test
43
44 clean:
45     $(MAKE) -C $(CLIENT_DIR) clean
46     $(MAKE) -C $(SERVER_DIR) clean
47
48 deploy:
49     $(MAKE) -C $(CLIENT_DIR) deploy
50     $(MAKE) -C $(SERVER_DIR) deploy
51     tools/createDeployable.sh

```

1.2 Tools für den Client

1.2.1 Makefile für Client

```

1 .PHONY: setup check format run test clean deploy
2
3 all:
4     npm run-script build
5
6 # install dependencies etc.
7 setup:
8     npm install
9
10 # run static analysis tools
11 check:
12     npm run-script eslint
13     npm run-script tslint
14
15 # fix linter errors automatically, if possible
16 fix:
17     npm run-script eslint-fix
18     npm run-script tslint-fix
19
20 format:
21     npm run-script format
22
23 # run project
24 run:
25     npm start
26
27 # run unit tests
28 test:
29     CI=true npm test
30
31 clean:
32     npm run-script clean
33
34 deploy:
35     npm run build

```

1.2.2 NPM Config

```
1  {
2    "name": "client",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "@fortawesome/fontawesome-svg-core": "^1.2.15",
7      "@fortawesome/free-regular-svg-icons": "^5.7.2",
8      "@fortawesome/free-solid-svg-icons": "^5.7.2",
9      "@fortawesome/react-fontawesome": "^0.1.4",
10     "@stomp/stompjs": "^5.2.0",
11     "@types/react-fontawesome": "^1.6.4",
12     "@types/react-notification-system": "^0.2.39",
13     "@types/react-select": "^2.0.15",
14     "@types/reactstrap": "^6.4.3",
15     "@types/socket.io-client": "^1.4.32",
16     "@types/sockjs-client": "^1.1.0",
17     "@types/uuid": "^3.4.4",
18     "ace-builds": "^1.4.2",
19     "bootstrap": "^4.1.3",
20     "conclave-lib": "file:externalLibs/conclave-lib",
21     "immutable": "^4.0.0-rc.12",
22     "jquery": "^3.3.1",
23     "mute-structs": "^0.5.4",
24     "popper.js": "^1.14.4",
25     "react": "^16.6.0",
26     "react-bootstrap": "^0.32.4",
27     "react-contextmenu": "^2.11.0",
28     "react-dom": "^16.6.0",
29     "react-notification-system": "^0.2.17",
30     "react-scripts": "^2.1.8",
31     "react-select": "^2.4.2",
32     "reactstrap": "^6.5.0",
33     "sockjs-client": "^1.3.0",
34     "stompjs-websocket": "^4.1.7"
35   },
36   "scripts": {
37     "start": "react-scripts start",
38     "build": "react-scripts build",
39     "test": "react-scripts test",
40     "eject": "react-scripts eject",
41     "clean": "rm -r build/*",
42     "eslint": "eslint -c .eslintrc.js src/**/*.{js,ts}",
43     "eslint-fix": "eslint -c .eslintrc.js --fix src/**/*.{js,ts}",
44     "tslint": "tslint -c tslint.json --project tsconfig.json 'src/**/*.{ts,js}'",
45     "tslint-fix": "tslint -c tslint.json --project tsconfig.json --fix 'src/**/*.{ts,js}'",
46     "format": "prettier --write 'src/**/*.{ts,js}'",
47     "tslint-config-check": "tslint-config-prettier-check ./tslint.json",
48     "eslint-config-check": "eslint --print-config . | eslint-config-prettier-check"
49   },
50   "eslintConfig": {
51     "extends": "react-app"
52   },
53   "browserslist": [
54     ">0.2%",
55     "not dead",
56     "not ie <= 11",
57     "not op_mini all"
58   ],
59   "devDependencies": {
60     "@fortawesome/fontawesome-free": "^5.5.0",
61     "@types/classnames": "^2.2.7",
62     "@types/jest": "^24.0.1",
63     "@types/node": "^10.12.10",
```

```
64      "@types/react": "^16.7.7",
65      "@types/react-dom": "^16.0.10",
66      "eslint": "^5.6.0",
67      "eslint-config-prettier": "^4.1.0",
68      "eslint-plugin-prettier": "^3.0.1",
69      "eslint-plugin-react": "^7.11.1",
70      "fast-check": "^1.10.0",
71      "neovim": "^4.2.1",
72      "prettier": "^1.15.2",
73      "react-fontawesome": "^1.6.1",
74      "react-test-renderer": "^16.6.3",
75      "tslint": "^5.14.0",
76      "tslint-config-airbnb": "^5.11.1",
77      "tslint-config-prettier": "^1.17.0",
78      "tslint-eslint-rules": "^5.4.0",
79      "tslint-plugin-prettier": "^2.0.1",
80      "tslint-react": "^3.6.0",
81      "typescript": "^3.3.3333"
82    }
83  }
```

1.2.3 TSLint

```
1  {
2    "defaultSeverity": "error",
3    "extends": [
4      "tslint:recommended",
5      "tslint-config-airbnb",
6      "tslint-react",
7      "tslint-eslint-rules",
8      "tslint-config-prettier",
9      "tslint-plugin-prettier"
10    ],
11    "jsRules": {},
12    "rules": {
13      "prettier": true,
14
15      "variable-name": [true, "ban-keywords", "check-format", "allow-leading-underscore"],
16      "comment-format": false,
17      "object-literal-shorthand": false,
18      "object-literal-sort-keys": false,
19      "interface-name": false,
20      "ordered-imports": false,
21      "no-else-after-return": false,
22
23      "member-ordering": [false],
24      "jsx-boolean-value": false,
25      "jsx-no-lambda": false,
26
27      "import-name": false,
28      "member-ordering": false,
29      "no-constant-condition": true,
30      "no-control-regex": true,
31      "no-duplicate-case": true,
32      "no-duplicate-imports": false,
33      "no-empty-character-class": true,
34      "no-ex-assign": true,
35      "no-extra-boolean-cast": true,
36      "no-inner-declarations": true,
37      "no-invalid-regexp": true,
```

```

38      "no-regex-spaces": true,
39      "ter-no-sparse-arrays": true,
40      "no-unexpected-multiline": true,
41      "valid-typeof": true,
42      "ter-no-proto": true,
43      "ter-no-script-url": true,
44      "ter-no-self-compare": true,
45      "handle-callback-err": true,
46      "array-bracket-spacing": false,
47      "ter-no-mixed-spaces-and-tabs": true,
48      "sort-imports": false,
49      "max-line-length": false,
50      "no-empty-interface": false,
51      "array-type": [true, "array"],
52      "no-console": false,
53      "increment-decrement": false,
54      "no-increment-decrement": false,
55      "no-plusplus": false
56    },
57    "rulesDirectory": []
58  }

```

1.2.4 ESLint

```

1  module.exports = {
2    "parser": "babel-eslint",
3    "env": {
4      "browser": true,
5      "es6": true,
6      // support test files
7      "jest": true
8    },
9    "plugins": ["prettier"],
10   "extends": [
11     "eslint:recommended",
12     "plugin:react/recommended",
13     "prettier",
14     "prettier/react"
15   ],
16   "parserOptions": {
17     "ecmaFeatures": {
18       "jsx": true
19     },
20     "ecmaVersion": 2018,
21     "sourceType": "module"
22   },
23   "settings": {
24     "react": {
25       "version": "15.0",
26     },
27   },
28   "rules": {
29     "prettier/prettier": "error",
30     "linebreak-style": [
31       "error",
32       "unix"
33     ],
34     "quotes": [
35       "error",
36       "single"

```

```
37     ],
38     // unused function arguments shall be no error
39     "no-unused-vars": ["error", { "args": "none" }],
40     "no-console": "off"
41   }
42 };
```

1.2.5 Prettier

```
1  {
2    "singleQuote": true,
3    "trailingComma": "es5"
4 }
```

1.3 Tools für den Server

1.3.1 Makefile für Server

```
1 .PHONY: setup check run test clean format deploy apidoc
2
3 all:
4   ./gradlew build
5
6 # install dependencies etc.
7 setup:
8
9 # run static analysis tools
10 check:
11   ./gradlew spotlessCheck
12   ./gradlew checkstyleMain
13   ./gradlew pmdMain
14   ./gradlew spotbugsMain
15
16 # auto format java code
17 format:
18   ./gradlew spotlessApply
19
20 # run project
21 run:
22   ./gradlew bootRun
23
24 # run unit and api tests
25 test:
26   ./helpers/api-test.sh
27
28 clean:
29   ./gradlew clean
30
```

```

31  deploy:
32      ./gradlew bootJar
33
34  apidoc:
35      ./gradlew asciidoctorPdf

```

1.3.2 Gradle Config für Server (build.gradle)

```

1 buildscript {
2     repositories {
3         jcenter()
4         mavenCentral()
5         maven {
6             url "https://plugins.gradle.org/m2/"
7         }
8     }
9     dependencies {
10         classpath "org.asciidoctor:asciidoctor-gradle-jvm:2.0.0-rc.1"
11         classpath "org.springframework.boot:spring-boot-gradle-plugin:2.0.5.RELEASE"
12         classpath "com.diffplug.spotless:spotless-plugin-gradle:3.21.1"
13         classpath "gradle.plugin.com.github.spotbugs:spotbugs-gradle-plugin:1.6.9"
14         classpath 'io.github.swagger2markup:swagger2markup-gradle-plugin:1.3.3'
15     }
16 }
17
18 apply plugin: 'java'
19 apply plugin: 'eclipse'
20 apply plugin: 'idea'
21 apply plugin: 'org.springframework.boot'
22 apply plugin: 'io.spring.dependency-management'
23
24
25 // coding style formatter and analysis
26 apply plugin: "com.diffplug.gradle.spotless"
27 apply plugin: 'checkstyle'
28
29 // static analysis tools
30 apply plugin: 'pmd'
31 apply plugin: "com.github.spotbugs"
32
33 // api doc generation
34 apply plugin: 'org.asciidoctor.jvm.convert'
35 apply plugin: "org.asciidoctor.jvm.pdf"
36 apply plugin: 'io.github.swagger2markup'
37
38 // import Key ant project
39 def keyFolder = "../key/key"
40 ant.importBuild "${keyFolder}/scripts/build.xml"
41 compileJava.dependsOn jarAll
42
43 bootJar {
44     baseName = 'KollaborierbaR'
45     version = '0.1.0'
46 }
47
48 repositories {
49     mavenCentral()
50 }
51
52 sourceSets {

```

```

53     test {
54         resources {
55             srcDir file('src/test/resources')
56             exclude '**/*.java'
57         }
58     }
59 }
60
61 sourceCompatibility = 1.8
62 targetCompatibility = 1.8
63
64 dependencies {
65     testCompile 'com.intuit.karate:karate-junit4:0.9.1'
66     testCompile 'com.intuit.karate:karate-apache:0.9.1'
67     //testCompile 'io.rest-assured:rest-assured:3.3.0'
68     implementation "org.eclipse.jdt:org.eclipse.jdt.core:3.15.0"
69     compile("org.springframework.boot:spring-boot-starter-web")
70     testCompile('org.springframework.boot:spring-boot-starter-test')
71     compile("org.springframework.boot:spring-boot-starter-websocket")
72     compile("org.javatuples:javatuples:1.2")
73     compile files('libs/logootsplit.jar')
74     compile('javax.xml.bind:jaxb-api')
75
76     // Database
77     compile("org.springframework.boot:spring-boot-starter-data-jpa")
78     compile("com.h2database:h2")
79
80     // Karate Test Framework for the server API
81     testCompile 'com.intuit.karate:karate-junit4:0.9.1'
82     testCompile 'com.intuit.karate:karate-apache:0.9.1'
83
84     // adds /actuator/health indicator, this way, other applications can check if the
85     // server is ready. Also useful for tests.
86     compile("org.springframework.boot:spring-boot-starter-actuator")
87
88     // include KeYnd KeY dependency libraries
89     compile fileTree(dir: "${keyFolder}/deployment/libs", include: "*.jar")
90     compile fileTree(dir: "${keyFolder}/deployment/components", include: "*.jar")
91
92     // SpringFox: Will automatically provide a HTML description of the API
93     compile "io.springfox:springfox-swagger2:2.9.2"
94     compile "io.springfox:springfox-swagger-ui:2.9.2"
95 }
96
97 wrapper {
98     gradleVersion = '4.10.2'
99 }
100
101 // Spotless coding style checker and formatter
102 spotless {
103     enforceCheck false // do not enforce formatting rules on regular build/check.
104     // ...instead call spotlessCheck / spotlessApply directly
105
106     java {
107         googleJavaFormat()
108     }
109 }
110
111 // Checkstyle tool. Also checks for aspects of the Google Java Style
112 checkstyle {
113     configFile rootProject.file('.checkstyle_google.xml')
114
115     ignoreFailures false
116     showViolations true
117     toolVersion ="8.18"
118

```

```

119     checkstyleTest.enabled = false
120 }
121
122 checkstyleMain {
123     configFile rootProject.file('.checkstyle_google.xml')
124     source ='src/main/java'
125 }
126
127 checkstyleTest {
128     configFile rootProject.file('.checkstyle_google.xml')
129     source ='src/test/java'
130 }
131
132 // PMD static analysis settings
133 // options: https://docs.gradle.org/current/dsl/org.gradle.api.plugins.quality.PmdExtension.html
134 pmd {
135     ruleSets = ["java-basic", "java-braces", "java-design"]
136     sourceSets = [sourceSets.main]
137     // only report problems where change is highly recommended.
138     // (on priority 3 there are some ridiculous warnings)
139     // see also https://pmd.sourceforge.io/pmd-5.3.3/customizing/rule-guidelines.html
140     rulePriority = 2
141     consoleOutput = true
142 }
143
144 // SpotBugs static analysis tool.
145 // Supports only Java 1.8 and 1.9
146 tasks.withType(com.github.spotbugs.SpotBugsTask) {
147     reports {
148         html {
149             enabled = true
150         }
151         xml {
152             enabled = false
153         }
154     }
155 }
156
157 spotbugs {
158     toolVersion = '3.1.9'
159     ignoreFailures = false
160     effort = 'max'
161     reportLevel = 'high'
162     sourceSets = [sourceSets.main, sourceSets.test]
163     excludeFilter = file("$rootProject.projectDir/.spotbugs-filter.xml")
164 }
165
166 // configure tests to show some actually useful debug information
167 test {
168     // https://docs.gradle.org/current/dsl/org.gradle.api.tasks.testing.logging.TestLoggingContainer.html
169     testLogging {
170         // set options for log level LIFECYCLE
171         events "failed"
172         exceptionFormat "full"
173
174         // set options for log level DEBUG
175         debug {
176             events "started", "skipped", "failed"
177             exceptionFormat "full"
178         }
179
180         // remove standard output/error logging from --info builds
181         // by assigning only 'failed' and 'skipped' events
182         info.events = ["failed", "skipped"]
183     }
184 }
```

```

185    // pull karate options into the runtime
186    systemProperty "karate.options", System.properties.getProperty("karate.options")
187    // pull karate env into the runtime
188    systemProperty "karate.env", System.properties.getProperty("karate.env")
189    // ensure tests are always run
190    outputs.upToDateWhen { false }
191 }
192
193 // API documentation generation
194 convertSwagger2markup {
195     swaggerInput "${projectDir}/specification/swagger.yaml"
196     outputDir file("${buildDir}/asciidoc/generated")
197     outputFile file("${buildDir}/asciidoc/generated/apidoc")
198     config = [
199         'swagger2markup.pathsGroupedBy' : 'TAGS',
200     ]
201 }
202
203 asciidoctorPdf {
204     version = 'Alpha 1.0'
205     dependsOn convertSwagger2markup
206     options doctype: 'report'
207     sourceDir file("${buildDir}/asciidoc/generated")
208
209     attributes [
210         toc: 'left',
211         toplevels: '3',
212         numbered: '',
213         sectlinks: '',
214         sectanchors: '',
215         hardbreaks: '',
216         sourceDir: file("${buildDir}/asciidoc/generated")
217     ]

```

1.3.3 Checkstyle

Wie im Qualitätssicherungsdokument beschrieben, wurde Checkstyle konfiguriert, sich an der Google Style Guide für Java Code zu orientieren. Die von Checkstyle zu diesem Zweck zur Verfügung gestellte Konfiguration¹ wurde daher verwendet. Kleine Anpassungen wurden vorgenommen um Kompatibilität mit dem ebenfalls eingesetzten Tool Spotless herzustellen. In anderen Fällen wurde der Grund kommentiert.

Spotless wurde ebenfalls für den Einsatz des Google Style Style konfiguriert (siehe `build.gradle`, 1.3.2). Wir setzen es zusätzlich zu Checkstyle ein, da es die automatische Formatierung des Code entsprechend der Guide erlaubt.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE module PUBLIC
3      "-//Checkstyle//DTD Checkstyle Configuration 1.3//EN"
4      "https://checkstyle.org/dtds/configuration_1_3.dtd">
5
6 <!--
7     Checkstyle configuration that checks the Google coding conventions from Google Java Style
8     that can be found at https://google.github.io/styleguide/javaguide.html.
9
10    Checkstyle is very configurable. Be sure to read the documentation at
11    http://checkstyle.sf.net (or in your downloaded distribution).
12
13    To completely disable a check, just comment it out or delete it from the file.
14
```

¹ https://checkstyle.org/google_style.html

```

15      Authors: Max Vetryenko, Ruslan Diachenko, Roman Ivanov.
16      -->
17
18  <module name="Checker">
19    <property name="charset" value="UTF-8"/>
20
21    <property name="severity" value="error"/>
22
23    <property name="fileExtensions" value="java, properties, xml"/>
24    <module name="FileTabCharacter">
25      <property name="eachLine" value="true"/>
26    </module>
27    <module name="TreeWalker">
28      <module name="OuterTypeFilename"/>
29      <module name="IllegalTokenText">
30        <property name="tokens" value="STRING_LITERAL, CHAR_LITERAL"/>
31        <property name="format"
32          value="\\u00(09|0(a|A)|0(c|C)|0(d|D)|22|27|5(C|c))|\\(0(10|11|12|14|15|42|47)|134)"/>
33        <property name="message"
34          value="Consider using special escape sequence instead of octal value or Unicode escaped
35            → value."/>
36      </module>
37      <module name="AvoidEscapedUnicodeCharacters">
38        <property name="allowEscapesForControlCharacters" value="true"/>
39        <property name="allowByTailComment" value="true"/>
40        <property name="allowNonPrintableEscapes" value="true"/>
41      </module>
42      <module name="LineLength">
43        <property name="max" value="100"/>
44        <property name="ignorePattern" value="^package.*|^import.*|a
45          → href|href|http://|https://|ftp://|&quot;[^&quot;]+&quot;"/>
46      </module>
47      <module name="AvoidStarImport"/>
48      <module name="OneTopLevelClass"/>
49      <module name="NoLineWrap"/>
50      <module name="EmptyBlock">
51        <property name="option" value="TEXT"/>
52        <property name="tokens"
53          value="LITERAL_TRY, LITERAL_FINALLY, LITERAL_IF, LITERAL_ELSE, LITERAL_SWITCH"/>
54      </module>
55      <module name="NeedBraces"/>
56      <module name="LeftCurly"/>
57      <module name="RightCurly">
58        <property name="id" value="RightCurlySame"/>
59        <property name="tokens"
60          value="LITERAL_TRY, LITERAL_CATCH, LITERAL_FINALLY, LITERAL_IF, LITERAL_ELSE,
61          LITERAL_DO"/>
62      </module>
63      <module name="RightCurly">
64        <property name="id" value="RightCurlyAlone"/>
65        <property name="option" value="alone"/>
66        <property name="tokens"
67          value="CLASS_DEF, METHOD_DEF, CTOR_DEF, LITERAL_FOR, LITERAL_WHILE, STATIC_INIT,
68          INSTANCE_INIT"/>
69      </module>
70      <module name="WhitespaceAround">
71        <property name="allowEmptyConstructors" value="true"/>
72        <property name="allowEmptyLambdas" value="true"/>
73        <property name="allowEmptyMethods" value="true"/>
74        <property name="allowEmptyTypes" value="true"/>
75        <property name="allowEmptyLoops" value="true"/>
76        <message key="ws.notFollowed"
77          value="WhitespaceAround: '{0}' is not followed by whitespace. Empty blocks may only be
78            → represented as '{}' when not part of a multi-block statement (4.1.3)"/>
79        <message key="ws.notPreceded"
80          value="WhitespaceAround: '{0}' is not preceded with whitespace."/>

```

```

78      </module>
79      <module name="OneStatementPerLine"/>
80      <module name="MultipleVariableDeclarations"/>
81      <module name="ArrayTypeStyle"/>
82      <module name="MissingSwitchDefault"/>
83      <module name="FallThrough"/>
84      <module name="UpperEll"/>
85      <module name="ModifierOrder"/>
86      <module name="EmptyLineSeparator">
87          <property name="allowNoEmptyLineBetweenFields" value="true"/>
88      </module>
89      <module name="SeparatorWrap">
90          <property name="id" value="SeparatorWrapDot"/>
91          <property name="tokens" value="DOT"/>
92          <property name="option" value="nl"/>
93      </module>
94      <module name="SeparatorWrap">
95          <property name="id" value="SeparatorWrapComma"/>
96          <property name="tokens" value="COMMA"/>
97          <property name="option" value="EOL"/>
98      </module>
99      <module name="SeparatorWrap">
100         <!-- ELLIPSIS is EOL until https://github.com/google/styleguide/issues/258 -->
101         <property name="id" value="SeparatorWrapEllipsis"/>
102         <property name="tokens" value="ELLIPSIS"/>
103         <property name="option" value="EOL"/>
104     </module>
105     <module name="SeparatorWrap">
106         <!-- ARRAY_DECLARATOR is EOL until https://github.com/google/styleguide/issues/259 -->
107         <property name="id" value="SeparatorWrapArrayDeclarator"/>
108         <property name="tokens" value="ARRAY_DECLARATOR"/>
109         <property name="option" value="EOL"/>
110     </module>
111     <module name="SeparatorWrap">
112         <property name="id" value="SeparatorWrapMethodRef"/>
113         <property name="tokens" value="METHOD_REF"/>
114         <property name="option" value="nl"/>
115     </module>
116     <module name="PackageName">
117         <property name="format" value="^ [a-z]+(\.[a-z][a-zA-Z]*)*$"/>
118         <message key="name.invalidPattern"
119             value="Package name '{0}' must match pattern '{1}'. "/>
120     </module>
121     <module name="TypeName">
122         <message key="name.invalidPattern"
123             value="Type name '{0}' must match pattern '{1}'. "/>
124     </module>
125     <module name="MemberName">
126         <property name="format" value="^ [a-z][a-zA-Z]*[a-zA-Z0-9]*$"/>
127         <message key="name.invalidPattern"
128             value="Member name '{0}' must match pattern '{1}'. "/>
129     </module>
130     <module name="ParameterName">
131         <property name="format" value="^ [a-z]([a-zA-Z][a-zA-Z0-9]*)?$"/>
132         <message key="name.invalidPattern"
133             value="Parameter name '{0}' must match pattern '{1}'. "/>
134     </module>
135     <module name="CyclomaticComplexity">
136         <property name="max" value="10"/>
137     </module>
138     <module name="LambdaParameterName">
139         <property name="format" value="^ [a-z]([a-zA-Z][a-zA-Z0-9]*)?$"/>
140         <message key="name.invalidPattern"
141             value="Lambda parameter name '{0}' must match pattern '{1}'. "/>
142     </module>
143     <module name="CatchParameterName">
```

```

144      <property name="format" value="^ [a-z] ([a-zA-Z0-9]* )? $" />
145      <message key="name.invalidPattern"
146          value="Catch parameter name '{0}' must match pattern '{1}'. "/>
147  </module>
148  <module name="LocalVariableName">
149      <property name="tokens" value="VARIABLE_DEF"/>
150      <property name="format" value="^ [a-z] ([a-zA-Z0-9]* )? $" />
151      <message key="name.invalidPattern"
152          value="Local variable name '{0}' must match pattern '{1}'. "/>
153  </module>
154  <module name="ClassTypeParameterName">
155      <property name="format" value="( ^ [A-Z] [0-9]? ) $ | ([A-Z] [a-zA-Z0-9]* [T] $) />
156      <message key="name.invalidPattern"
157          value="Class type name '{0}' must match pattern '{1}'. "/>
158  </module>
159  <module name="MethodTypeParameterName">
160      <property name="format" value="( ^ [A-Z] [0-9]? ) $ | ([A-Z] [a-zA-Z0-9]* [T] $) />
161      <message key="name.invalidPattern"
162          value="Method type name '{0}' must match pattern '{1}'. "/>
163  </module>
164  <module name="InterfaceTypeParameterName">
165      <property name="format" value="( ^ [A-Z] [0-9]? ) $ | ([A-Z] [a-zA-Z0-9]* [T] $) />
166      <message key="name.invalidPattern"
167          value="Interface type name '{0}' must match pattern '{1}'. "/>
168  </module>
169  <module name="NoFinalizer"/>
170  <module name="GenericWhitespace">
171      <message key="ws.followed"
172          value="GenericWhitespace '{0}' is followed by whitespace."/>
173      <message key="ws.preceded"
174          value="GenericWhitespace '{0}' is preceded with whitespace."/>
175      <message key="ws.illegalFollow"
176          value="GenericWhitespace '{0}' should followed by whitespace."/>
177      <message key="ws.notPreceded"
178          value="GenericWhitespace '{0}' is not preceded with whitespace."/>
179  </module>
180  <module name="Indentation">
181      <property name="basicOffset" value="2"/>
182      <property name="braceAdjustment" value="0"/>
183      <property name="caseIndent" value="2"/>
184      <property name="throwsIndent" value="4"/>
185      <property name="lineWrappingIndentation" value="4"/>
186      <property name="arrayInitIndent" value="2"/>
187  </module>
188  <module name="AbbreviationAsWordInName">
189      <property name="ignoreFinal" value="false"/>
190      <property name="allowedAbbreviationLength" value="1"/>
191  </module>
192  <module name="OverloadMethodsDeclarationOrder"/>
193  <module name="VariableDeclarationUsageDistance"/>
194  <module name="CustomImportOrder">
195      <property name="sortImportsInGroupAlphabetically" value="true"/>
196      <property name="separateLineBetweenGroups" value="true"/>
197      <property name="customImportOrderRules" value="STATIC###THIRD_PARTY_PACKAGE"/>
198  </module>
199  <module name="MethodParamPad"/>
200  <module name="NoWhitespaceBefore">
201      <property name="tokens"
202          value="COMMA, SEMI, POST_INC, POST_DEC, DOT, ELLIPSIS, METHOD_REF"/>
203      <property name="allowLineBreaks" value="true"/>
204  </module>
205  <module name="ParenPad"/>
206  <module name="OperatorWrap">
207      <property name="option" value="NL"/>
208      <property name="tokens"
209          value="BAND, BOR, BSR, BXOR, DIV, EQUAL, GE, GT, LAND, LE, LITERAL_INSTANCEOF, LOR,
```

```

210             LT, MINUS, MOD, NOT_EQUAL, PLUS, QUESTION, SL, SR, STAR, METHOD_REF "/>
211         </module>
212         <module name="AnnotationLocation">
213             <property name="id" value="AnnotationLocationMostCases"/>
214             <property name="tokens"
215                 value="CLASS_DEF, INTERFACE_DEF, ENUM_DEF, METHOD_DEF, CTOR_DEF"/>
216         </module>
217         <module name="AnnotationLocation">
218             <property name="id" value="AnnotationLocationVariables"/>
219             <property name="tokens" value="VARIABLE_DEF"/>
220             <property name="allowSamelineMultipleAnnotations" value="true"/>
221         </module>
222         <module name="NonEmptyAtclauseDescription"/>
223         <module name="JavadocTagContinuationIndentation"/>
224         <!-- Bei manchen Methoden ist ein Summary nicht notwendig, da z. B. der return tag bereits alles
225             erklärt
226         <module name="SummaryJavadoc">
227             <property name="forbiddenSummaryFragments"
228                 value="^@return the */^This method returns |^A [{}@code [a-zA-Z0-9]+[]]( is a )"/>
229         </module>
230         -->
231         <module name="JavadocParagraph"/>
232         <module name="AtclauseOrder">
233             <property name="tagOrder" value="@param, @return, @throws, @deprecated"/>
234             <property name="target"
235                 value="CLASS_DEF, INTERFACE_DEF, ENUM_DEF, METHOD_DEF, CTOR_DEF, VARIABLE_DEF"/>
236         </module>
237         <!-- Bei manchen Methoden (getter, setter) ist JavaDoc nicht sinnvoll.
238         <module name="JavadocMethod">
239             <property name="scope" value="public"/>
240             <property name="allowMissingParamTags" value="true"/>
241             <property name="allowMissingThrowsTags" value="true"/>
242             <property name="allowMissingReturnTag" value="true"/>
243             <property name="minLineCount" value="2"/>
244             <property name="allowedAnnotations" value="Override, Test"/>
245             <property name="allowThrowsTagsForSubclasses" value="true"/>
246         </module>
247         -->
248         <module name="MethodName">
249             <property name="format" value="^[a-z][a-z0-9][a-zA-Z0-9_]*$"/>
250             <message key="name.invalidPattern"
251                 value="Method name '{0}' must match pattern '{1}'. "/>
252         </module>
253         <module name="SingleLineJavadoc">
254             <property name="ignoredTags" value="@return"/>
255             <property name="ignoreInlineTags" value="true"/>
256         </module>
257         <module name="EmptyCatchBlock">
258             <property name="exceptionVariableName" value="expected"/>
259         </module>
260         <module name="CommentsIndentation"/>
261     </module>

```

1.3.4 Spotbugs

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <FindBugsFilter>
3      <Match>
4          <Class name="synchronization.data.Identifier"/>

```

```

5      <Bug pattern="NM_SAME_SIMPLE_NAME_AS_SUPERCLASS"/>
6  </Match>
7  <Match>
8      <Class name="synchronization.data.IdentifierInterval"/>
9      <Bug pattern="NM_SAME_SIMPLE_NAME_AS_SUPERCLASS"/>
10 </Match>
11 <Match>
12     <Class name="synchronization.data.LogootSAdd"/>
13     <Bug pattern="NM_SAME_SIMPLE_NAME_AS_SUPERCLASS"/>
14 </Match>
15 <Match>
16     <Class name="synchronization.data.LogootSDel"/>
17     <Bug pattern="NM_SAME_SIMPLE_NAME_AS_SUPERCLASS"/>
18 </Match>
19 </FindBugsFilter>

```

1.4 Beispieldurchführung

Es folgt eine Ausführung sämtlicher Schritte des Buildprozesses, wie sie mit `make pipeline` durchgeführt werden kann:

```

1  anton@ArchSurface % make pipeline
2  if [ -z "" ]; then \
3      git submodule update --init --recursive; \
4  fi;
5  make -C client setup
6  make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/cli'
7  npm install
8  npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules/fsevents):
9  npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {
10 npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node_modules/chokidar/node_modu
11 npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {
12
13 audited 36942 packages in 10.775s
14 found 63 low severity vulnerabilities
15   run `npm audit fix` to fix them, or `npm audit` for details
16 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/clie
17 make -C server setup
18 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/ser
19 make[1]: Nothing to be done for 'setup'.
20 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/serv
21 make -C client check
22 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/cli
23 npm run-script eslint
24
25 > client@0.1.0 eslint /home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/client
26 > eslint -c .eslintrc.js src/**/*.js*
27
28 npm run-script tslint
29
30 > client@0.1.0 tslint /home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/client
31 > tslint -c tslint.json --project tsconfig.json 'src/**/*.ts'
32
33 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/clie
34 make -C server check
35 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/ser
36 ./gradlew spotlessCheck
37 Starting a Gradle Daemon, 17 busy Daemons could not be reused, use --status for details
38 > Task :spotlessJava UP-TO-DATE
39 > Task :spotlessJavaCheck UP-TO-DATE
40 > Task :spotlessCheck UP-TO-DATE

```

```

41
42 BUILD SUCCESSFUL in 10s
43 1 actionable task: 1 up-to-date
44 ./gradlew checkstyleMain
45
46 > Task :jarAll
47 [ant:echo] Please set (if you have not yet done so) the environment variable ANT_OPTS to -Xms5
48
49 > Task :compileJava UP-TO-DATE
50 > Task :processResources UP-TO-DATE
51 > Task :classes UP-TO-DATE
52 > Task :checkstyleMain UP-TO-DATE
53
54 BUILD SUCCESSFUL in 6s
55 4 actionable tasks: 1 executed, 3 up-to-date
56 ./gradlew pmdMain
57
58 > Task :jarAll
59 [ant:echo] Please set (if you have not yet done so) the environment variable ANT_OPTS to -Xms5
60
61 > Task :compileJava UP-TO-DATE
62 > Task :processResources UP-TO-DATE
63 > Task :classes UP-TO-DATE
64 > Task :pmdMain UP-TO-DATE
65
66 BUILD SUCCESSFUL in 4s
67 4 actionable tasks: 1 executed, 3 up-to-date
68 ./gradlew spotbugsMain
69
70 > Task :jarAll
71 [ant:echo] Please set (if you have not yet done so) the environment variable ANT_OPTS to -Xms5
72
73 > Task :compileJava UP-TO-DATE
74 > Task :processResources UP-TO-DATE
75 > Task :classes UP-TO-DATE
76 > Task :spotbugsMain UP-TO-DATE
77
78 BUILD SUCCESSFUL in 3s
79 4 actionable tasks: 1 executed, 3 up-to-date
80 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/serv
81 make -C client
82 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/cli
83 npm run-script build
84
85 > client@0.1.0 build /home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/client
86 > react-scripts build
87
88 Creating an optimized production build...
89 Compiled successfully.
90
91 File sizes after gzip:
92
93   265.63 KB  build/static/js/2.c3d0d137.chunk.js
94   34.42 KB   build/static/css/2.938089cc.chunk.css
95   22.8 KB    build/static/js/main.26128f75.chunk.js
96   9.07 KB   build/static/css/main.f7a6a185.chunk.css
97   762 B     build/static/js/runtime~main.a8a9905a.js
98
99 The project was built assuming it is hosted at the server root.
100 You can control this with the homepage field in your package.json.
101 For example, add this to build it for GitHub Pages:
102
103   "homepage" : "http://myname.github.io/myapp",
104
105 The build folder is ready to be deployed.
106 You may serve it with a static server:

```

```

107
108     npm install -g serve
109     serve -s build
110
111 Find out more about deployment here:
112
113     https://bit.ly/CRA-deploy
114
115 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/clie
116 make -C client test
117 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/cli
118 CI=true npm test
119
120 > client@0.1.0 test /home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/client
121 > react-scripts test
122
123 PASS src/__tests__/file-icon-tests.jsx
124 PASS src/__tests__/dummy.js
125 PASS src/__tests__/markerList.ts
126
127 Test Suites: 3 passed, 3 total
128 Tests:      3 passed, 3 total
129 Snapshots:   0 total
130 Time:       6.004s
131 Ran all test suites.
132 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/clie
133 make -C server test
134 make[1]: Entering directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/ser
135 ./helpers/api-test.sh
136 Preparing tests on server api...
137 Building server beforehand...
138
139 > Task :jarAll
140 [ant:echo] Please set (if you have not yet done so) the environment variable ANT_OPTS to -Xms5
141
142 > Task :compileJava UP-TO-DATE
143 > Task :processResources UP-TO-DATE
144 > Task :classes UP-TO-DATE
145 > Task :bootJar
146 > Task :jar SKIPPED
147 > Task :assemble
148 > Task :checkstyleMain UP-TO-DATE
149 > Task :compileTestJava UP-TO-DATE
150 > Task :processTestResources UP-TO-DATE
151 > Task :testClasses UP-TO-DATE
152 > Task :checkstyleTest SKIPPED
153 > Task :pmdMain UP-TO-DATE
154 > Task :spotbugsMain UP-TO-DATE
155 > Task :spotbugsTest UP-TO-DATE
156 > Task :check
157 > Task :build
158
159 BUILD SUCCESSFUL in 5s
160 10 actionable tasks: 2 executed, 8 up-to-date
161 Running server.
162 Waiting for server to be ready...
163 Server is ready! Running api tests...
164 Starting a Gradle Daemon, 18 busy Daemons could not be reused, use --status for details
165
166 > Task :jarAll
167 [ant:echo] Please set (if you have not yet done so) the environment variable ANT_OPTS to -Xms5
168
169 > Task :compileJava UP-TO-DATE
170 > Task :processResources UP-TO-DATE
171 > Task :classes UP-TO-DATE
172 > Task :compileTestJava UP-TO-DATE

```

```
173 > Task :processTestResources UP-TO-DATE
174 > Task :testClasses UP-TO-DATE
175 > Task :test
176
177 BUILD SUCCESSFUL in 1m 0s
178 6 actionable tasks: 2 executed, 4 up-to-date
179 make[1]: Leaving directory '/home/anton/Documents/Studium/WiSe18_19/BP/Code/kollaborierbar/serv
180 make pipeline 92.60s user 6.40s system 62% cpu 2:39.22 total
```

KollaborierbaR Server HTTP API

Version Alpha 1.0

Table of Contents

1. Overview	1
1.1. Version information.....	1
1.2. Contact information.....	1
1.3. URI scheme.....	1
1.4. Tags	1
2. Resources	1
2.1. Linter-controller	1
2.1.1. Run static analysis on a java source file	1
2.2. Project-controller	2
2.2.1. List projects stored on server.....	2
2.2.2. Load content and name of specific file on the server	3
2.2.3. Load file tree of a specific project stored on the server.....	3
2.2.4. Deletes a project	4
2.2.5. Update the contents of a file stored the server or rename it.....	4
2.2.6. Create a new file or folder on the server.....	5
2.2.7. Delete a file or folder from a project	6
2.3. Proof-controller.	7
2.3.1. Prove obligations within a Java file using KeY	7
2.3.2. Lists proof obligation indices of saved proofs.....	8
2.3.3. Add a proof result to the server history.....	9
2.3.4. List ids of proofs saved to history.....	10
2.3.5. Retrieve a proof from the server history.....	11
2.3.6. Delete a proof result from the server history.....	12
2.3.7. Save a temporary proof result.....	13
2.3.8. Retrieve the latest temporary proof result.	14
3. Definitions	14
3.1. Diagnostic	14
3.2. FileOrFolderItem	15
3.3. FileUpdateData	15
3.4. ObligationResult	15
3.5. OpenGoalInfo	16
3.6. OpenedFileResponse	16
3.7. ProofNode	17
3.8. ProofResult	18

1. Overview

This is the central server of the KollaborierbaR collaborative Java specification and verification web app.

1.1. Version information

Version : 0.1

1.2. Contact information

Contact Email : kollaborierbar-team@googlegroups.com

1.3. URI scheme

Host : localhost:9000

BasePath : /

Schemes : HTTP

1.4. Tags

- linter-controller : Generate diagnostics for Java code by static analysis
- project-controller : Access projects stored on server
- proof-controller : Generate proofs for Java code using KeY

2. Resources

2.1. Linter-controller

Generate diagnostics for Java code by static analysis

2.1.1. Run static analysis on a java source file

POST /lint

Description

Runs a static analysis on a java source file. This includes the static analysis framework of the Eclipse JDT and tests for the use of Java features, which are not yet supported by the automatic prover of the KeY Project.

Will return an empty list, if static analysis fails, or if no source code was submitted.

Parameters

Type	Name	Description	Schema
Query	name <i>required</i>	Filename of the uploaded source code	string
Body	source <i>required</i>	Java source code to be analyzed	string

Responses

HTTP Code	Description	Schema
200	OK. Successful static analysis. Returning results	< Diagnostic > array

Consumes

- [application/json](#)

Produces

- [application/json; charset=UTF-8](#)

2.2. Project-controller

Access projects stored on server

2.2.1. List projects stored on server

```
GET /projects
```

Responses

HTTP Code	Description	Schema
200	OK. A list of project names	< string > array

Produces

- [application/json; charset=UTF-8](#)

Example HTTP response

```
A.120
```

Response 200

```
[ "My Project", "KeY" ]
```

2.2.2. Load content and name of specific file on the server

```
GET /projects/**
```

Description

Replace ****** with any file path, starting with the project name as root directory.

For example: [/projects/SimpleJML/src/examples/IntegerUtil.java](#)

Responses

HTTP Code	Description	Schema
200	OK	OpenedFileResponse
404	Not Found. The requested file could not be found within the project.	string
500	Internal server error. The file could not be read, though it exists.	string

Produces

- [*/*](#)

2.2.3. Load file tree of a specific project stored on the server

```
GET /projects/{projectname}
```

Parameters

Type	Name	Description	Schema
Path	projectname <i>required</i>	Name of the project, for which the file tree shall be retrieved.	string

Responses

HTTP Code	Description	Schema
200	OK. Tree file structure of a (java) project. Consists of files and folders. The name of the root folder is the same as the project name. (The data structure is recursive, which OpenAPI 2.0 descriptions can not express, therefore the specified model lacks the recursion.)	FileOrFolderItem
500	Internal Server Error. Occurs, if project does not exist.	No Content

Produces

- `application/json; charset=UTF-8`

2.2.4. Deletes a project

```
DELETE /projects/{projectname}
```

Parameters

Type	Name	Description	Schema
Path	projectname <i>required</i>	Name of the project which shall be deleted.	string

Responses

HTTP Code	Description	Schema
200	OK. Project got successfully deleted.	object
404	Not found. There exists no such project on the server.	string
500	Internal Server Error. The project exists, but could not be deleted.	string

Produces

- `application/json; charset=UTF-8`

2.2.5. Update the contents of a file stored the server or rename it.

```
POST /projects/{projectname}/**
```

Description

1. Allows to save a file, such that other clients may access it using the server API.
2. Also allows to rename a file.

If the field `fileContent` is not set in the body, case 2 applies.

Otherwise, if `fileContent` is set, the content of the file will be replaced with the contents of `fileContent`, such that all subsequent calls to GET `/projects/<projectname>/<path-to-file>` will return the updated content. In this case, the file name will not be changed, even if `fileName` is set, too. It is assumed, that the file content is encoded in UTF-8.

Replace `**` with a path to a file within the specified project.

For example: `src/example/IntegerUtils.java`

Parameters

Type	Name	Description	Schema
Path	<code>projectname</code> <i>required</i>	Name of the project of which we'd like to change the content of a file.	string
Body	<code>updateData</code> <i>required</i>	Name and new content	<code>FileUpdateData</code>

Responses

HTTP Code	Description	Schema
<code>200</code>	OK. The change was successfully applied. The new project tree is returned, see <code>GET /projects/{projectname}</code> . If the file name has not been changed, the returned tree will also appear unchanged, given no other client changed the project structure in the meantime.	<code>FileOrFolderItem</code>
<code>400</code>	Bad request. The file does not exist, or the rename path is invalid. Can also be caused by an internal error while updating.	string

Consumes

- `application/json; charset=UTF-8`

Produces

- `application/json; charset=UTF-8`

2.2.6. Create a new file or folder on the server.

```
PUT /projects/{projectname}/**
```

Description

Allows to create a file or folder on the server, such that other clients may access it using the server API.

Replace ****** with a path within the specified project.

For example: [src/example/IntegerUtils.java](#)

Parameters

Type	Name	Description	Schema
Path	projectname <i>required</i>	Name of the project, within we want to create a new file/folder	string
Query	type <i>required</i>	Indicates, whether to create a file or folder. Possible values: 'file', 'folder'	string

Responses

HTTP Code	Description	Schema
200	OK. The file/folder was successfully created. The updated project tree is returned, see GET /projects/{projectname} .	FileOrFolderItem
400	Bad request. Sent in the following cases: 1. There already exists a file/folder at the given path. 2. The type parameter has an invalid value. It must equal either 'file' or 'folder'	string

Consumes

- [application/json](#)

Produces

- [application/json; charset=UTF-8](#)

2.2.7. Delete a file or folder from a project

```
DELETE /projects/{projectname}/**
```

Parameters

Type	Name	Description	Schema
Path	projectname <i>required</i>	Name of the project, where a file/folder shall be deleted	string

Responses

HTTP Code	Description	Schema
200	OK. The file/folder was successfully deleted. The updated project tree is returned, see also GET /projects/{projectname} .	
404	Not found. Returned, if the specified file/folder does not exist or the path is invalid.	string
500	Internal server error. The file/folder which shall be deleted does exist, but could not be deleted.	string

Produces

- [application/json; charset=UTF-8](#)

2.3. Proof-controller

Generate proofs for Java code using KeY

2.3.1. Prove obligations within a Java file using KeY

```
GET /proof/**/{className}.java
```

Description

This method will try to prove the specified proof obligations within the given file using KeY (<https://www.key-project.org/>)

If no obligation indices are specified, the server will try to prove all obligations within the given file.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class in which the proofs shall be run.	string

Type	Name	Description	Schema
Query	macro <i>optional</i>	Path to the macro file to use for the requested proof(s), including the project name. Can be omitted if the proof should be executed without a macro/ proof script.	string
Query	obligationIdxs <i>optional</i>	Indices of the proof obligations, which shall be proven, counted from the beginning of the Java file.	< integer (int32) > array

Responses

HTTP Code	Description	Schema
200	OK. If an obligation could not be proven, the reasons are encoded within the returned structure. Beware of the example return value generated by Swagger, which can for example contain successful proofs in the failed list. This is because OpenAPI 2.0's way of specifying examples is very restricted.	ProofResult

Produces

- `application/json; charset=UTF-8`

2.3.2. Lists proof obligation indices of saved proofs.

```
GET /proof/**/{className}.java/obligation
```

Description

Lists all obligation indices for the given file, for which there are saved proofs available on the server.

The indices are counted per file from top to bottom for each discovered proof obligation.

The returned indices can be used to retrieve saved proofs using [GET /proof//{className}.java/obligation/{obligation index}/](#).

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to list obligation indices of saved proofs.	string

Responses

HTTP Code	Description	Schema
200	OK. This method will always return code 200. If there are no saved proofs, an empty list will be returned.	< integer (int32) > array

Produces

- application/json; charset=UTF-8

Example HTTP response

Response 200

```
[ 0, 2, 3 ]
```

2.3.3. Add a proof result to the server history.

```
POST /proof/**/{className}.java/obligation/{obligationIdx}/history
```

Description

The server can save proof results within a history for each proof obligation, for each file, such that they can be retrieved and reviewed at a later time.

Using this route, a proof result can be added to the history.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to save a proof result.	string
Path	obligationIdx <i>required</i>	Index of the proof obligation the saved proof belongs to, counted from top to bottom in the given Java file.	integer (int32)
Body	obligationResult <i>required</i>	Proof result, which shall be saved.	ObligationResult

Responses

HTTP Code	Description	Schema
200	OK. Returns the unique history id of the saved proof.	integer (int32)

Consumes

- application/json

Produces

- application/json; charset=UTF-8

Example HTTP response

Response 200

```
3
```

2.3.4. List ids of proofs saved to history

```
GET /proof/**/{className}.java/obligation/{obligationIdx}/history
```

Description

The server can save proof results within a history, such that they can be retrieved and reviewed at a later time.

Each saved proof result is given a history id, which can be used to retrieve a result using [GET /proof/**/{className}.java/obligation/{obligationIdx}/history/{history id}](#).

The id also decodes the age of a saved result, more recent proof results get assigned bigger ids than their predecessors.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to retrieve proof history information.	string
Path	obligationIdx <i>required</i>	A history is maintained for each proof obligation. This is the index of the obligation counted from the top of the file, for which proof result history information shall be retrieved.	integer (int32)

Responses

HTTP Code	Description	Schema
200	OK. This method will always return code 200. If there are no saved proofs, an empty list will be returned.	< integer (int32) > array

Produces

- application/json; charset=UTF-8

Example HTTP response

Response 200

```
[ 2, 20, 3 ]
```

2.3.5. Retrieve a proof from the server history.

```
GET /proof/**/{className}.java/obligation/{obligationIdx}/history/{historyIdx}
```

Description

The server can save proof results within a history for each proof obligation, for each file, such that they can be retrieved and reviewed at a later time.

Using this route, a proof result can be retrieved from the history.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to retrieve a proof result.	string
Path	historyIdx <i>required</i>	Unique history id of the saved proof. See GET /proof/**/{className}.java/obligation/{obligationIdx}/history	integer (int32)
Path	obligationIdx <i>required</i>	Index of the proof obligation the proof which shall be retrieved belongs to, counted from top to bottom in the given Java file.	integer (int32)

Responses

HTTP Code	Description	Schema
200	OK. Returns the saved proof result.	ObligationResult
404	Not found. There is no proof result saved for the given path and parameters.	No Content

Produces

- application/json; charset=UTF-8

2.3.6. Delete a proof result from the server history.

```
DELETE /proof/**/{className}.java/obligation/{obligationIdx}/history/{historyIdx}
```

Description

The server can save proof results within a history for each proof obligation, for each file, such that they can be retrieved and reviewed at a later time.

Using this route, a proof result can be deleted from the history.

Parameters

Type	Name	Description	Schema
Path	className required	Name of the class, for which we want to delete a proof result from the history.	string
Path	historyIdx required	Unique history id of the proof result which shall be deleted from the history. See GET /proof/**/{className}.java/obligation/{obligationIdx}/history	integer (int32)
Path	obligationIdx required	Index of the proof obligation the proof which shall be deleted belongs to, counted from top to bottom in the given Java file.	integer (int32)

Responses

HTTP Code	Description	Schema
200	OK. Successfully deleted proof result from history.	No Content
404	Not found. The specified proof result can not be deleted from history, since there is no such result saved.	No Content

Produces

- application/json; charset=UTF-8

2.3.7. Save a temporary proof result.

```
POST /proof/**/{className}.java/obligation/{obligationIdx}/last
```

Description

In addition to a permanent proof result history, the server can temporarily store the latest prove result. It is lost as soon as the server is shut down.

This feature is used by KollaborierbaR to always share the latest proof result by any developer with everyone working on the same file.

Using this route, such a temporarily result can be uploaded.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to save a temporary proof result.	string
Path	obligationIdx <i>required</i>	Index of the proof obligation to which the uploaded result belongs, counted from top to bottom in the given Java file.	integer (int32)
Body	obligationResult <i>required</i>	proof result to be stored	ObligationResult

Responses

HTTP Code	Description	Schema
200	OK. The proof result has been temporarily stored.	No Content

Consumes

- application/json

Produces

- application/json; charset=UTF-8

2.3.8. Retrieve the latest temporary proof result.

```
GET /proof/**/{className}.java/obligation/{obligationIdx}/last
```

Description

In addition to a permanent proof result history, the server can temporarily store the latest prove result. It is lost as soon as the server is shut down.

This feature is used by KollaborierbaR to always share the latest proof result by any developer with everyone working on the same file.

Using this route, this temporarily stored result can be retrieved.

Parameters

Type	Name	Description	Schema
Path	className <i>required</i>	Name of the class, for which we want to retrieve the latest, temporary proof result.	string
Path	obligationIdx <i>required</i>	Index of the proof obligation for which the latest temporary results shall be retrieved, counted from top to bottom in the given Java file.	integer (int32)

Responses

HTTP Code	Description	Schema
200	OK. Returns the temporarily saved proof result.	ObligationResult
404	Not found. There has no temporary proof result been saved for the given path and parameters.	No Content

Produces

- application/json; charset=UTF-8

3. Definitions

3.1. Diagnostic

Name	Description	Schema
endCol <i>optional</i>	Example : 42	integer (int64)

A.132

Name	Description	Schema
endRow <i>optional</i>	Example : 24	integer (int64)
kind <i>optional</i>		enum (ERROR, WARNING, NOTE, NOT_SUPPORTED)
message <i>optional</i>	Example : "Syntax error"	string
startCol <i>optional</i>	Example : 0	integer (int64)
startRow <i>optional</i>	Example : 24	integer (int64)

3.2. FileOrFolderItem

Name	Description	Schema
contents <i>optional</i>	Contents of the folder. This field is not set, if the item is a file, not a folder.	< FileOrFolderItem > array
name <i>required</i>	Example : "Main.java"	string
type <i>required</i>		enum (file, folder)

3.3. FileUpdateData

Name	Description	Schema
fileContent <i>optional</i>	Example : "public class Main {\n public static void main(String[] args) {}}\n"	string
fileName <i>optional</i>	Example : "/projects/src/main/java/Main.java"	string

3.4. ObligationResult

Name	Description	Schema
id <i>optional</i>	Unique id used to identify obligation results in other routes	number
kind <i>required</i>		enum (success, failure, error)
obligationIdx <i>required</i>	Obligation index this proof belongs to. It is counted from top to bottom in the corresponding source file. Example : 0	integer (int32)
openGoals <i>required</i>		< OpenGoalInfo > array
proofTree <i>optional</i>		ProofNode
resultMsg <i>required</i>	Human readable statement about the result state of the proof. Example : "Contract 'JML normal_behavior operation contract 0' of IntegerUtil2::add is verified."	string
targetName <i>required</i>	Target subject of the proof, for example a method name. Example : "IntegerUtil2::add"	string

3.5. OpenGoalInfo

Name	Description	Schema
formula <i>required</i>	Created by calling <code>toString</code> on KeY goals, may not be equal to <code>sequent</code> , but oftentimes is. Example : "wellFormed(heap), measuredByEmpty => y = 0\n"	string
id <i>required</i>	Example : 64	integer (int64)
sequent <i>required</i>	Created using the <code>quickPrintSequent</code> function of KeY's <code>LogicPrinter</code> . May not be equal to <code>formula</code> . Example : "wellFormed(heap), measuredByEmpty => y = 0\n"	string

3.6. OpenedFileResponse

Name	Description	Schema
fileName <i>optional</i>	Example : "IntegerUtil.java"	string
fileText <i>optional</i>	Example : "public class IntegerUtil { @ normal_behavior @ ensures \result == x + y; public static int add(int x, int y) { return x + y; } @ normal_behavior @ requires (x != y); @ ensures \result == x - y; @ exceptional_behaviour @ requires (x == y); @ signals (RuntimeException) true; public static int sub(int x, int y) { if (x == y) { throw new RuntimeException();} return x + y; }"	string

3.7. ProofNode

Name	Description	Schema
children <i>optional</i>		< ProofNode > array
kind <i>optional</i>		enum (ClosedProofTree, OneStepSimplification, OpenProofTree, BranchNode, OpenGoal, ClosedGoal, InteractiveGoal, LinkedGoal, DefaultNode)
oneStepId <i>optional</i>	Unique id of this node relative to all its siblings, if it is a child of a One Step Simplification node. Otherwise, it always equals 0. Together with <code>serialNr</code> , this field forms a unique id within the whole proof tree. Example : 0	integer (int32)
sequent <i>optional</i>	Example : " \n=>\n wellFormed(heap)\n & ..."	string

Name	Description	Schema
serialNr <i>optional</i>	Unique id of this node within the proof tree, if its parent is not a One Step Simplification. If its parent is a One Step simplification, it forms a unique key together with oneStepId . Example : 0	integer (int32)
text <i>optional</i>	Example : "IntegerUtil2[IntegerUtil2::add(int,int)].JML normal_behavior operation contract.0"	string

3.8. ProofResult

Name	Description	Schema
failed <i>optional</i>	Proofs, which could not be closed.	< ObligationResult > array
succeeded <i>optional</i>	Successful proofs.	< ObligationResult > array

Das folgende Listing enthält die Spezifikation als `.yaml` Datei. Mit dieser Datei kann die Spezifikation weiterentwickelt werden und gleichzeitig stellt sie ein maschinenlesbares Format dar, aus der beispielsweise die obige PDF Dokumentation der Spezifikation generiert wurde.

```
1  swagger: '2.0'
2  info:
3    description: >-
4      This is the central server of the KollaborierbaR collaborative Java
5      specification and verification web app.
6    version: '0.1'
7    title: KollaborierbaR Server HTTP API
8    contact:
9      email: "kollaborierbar-team@googlegroups.com"
10   host: 'localhost:9000'
11   basePath: /
12   tags:
13     - name: linter-controller
14       description: Generate diagnostics for Java code by static analysis
15     - name: project-controller
16       description: Access projects stored on server
17     - name: proof-controller
18       description: Generate proofs for Java code using KeY
19   schemes:
20     - http
21   paths:
22     /lint:
23       post:
24         tags:
25           - linter-controller
26           summary: Run static analysis on a java source file
27           description: >-
28             Runs a static analysis on a java source file.
29             This includes the static analysis framework of the Eclipse JDT and tests for the use of Java
30             features, which are not yet supported by the automatic prover of the KeY Project.
31
32           Will return an empty list, if static analysis fails, or if no source code was submitted.
33   operationId: lintUsingPOST
34   consumes:
35     - application/json
36   produces:
37     - application/json;charset=UTF-8
38   parameters:
39     - name: name
40       in: query
41       description: Filename of the uploaded source code
42       required: true
43       type: string
44     - in: body
45       name: source
46       description: Java source code to be analyzed
47       required: true
48       schema:
49         type: string
```

```

50
example: "public class LimitedIntegerSet {\n\t//@ public invariant (\\forall int i,j; i>=0 &&
51   i<j && j<size; arr[i] != arr[j]);\n\tprivate /*@ spec_public @*/ int[] arr;\n\t\\n\\t\\n\\t//@
52   public invariant size >= 0 && size <= arr.length;\n\tprivate /*@ spec_public @*/ int
53   size;\n\\n\\tpublic LimitedIntegerSet(int limit) {\n\t\\tthis.arr = new
54   int[limit];\n\t\\n\\t/*@ public normal_behavior //This is a JML comment!\n      @
55   ensures \\result == (\\exists int i;\n          @\n          0 <= i && i <
56   size;\n          @\n          arr[i] == elem);\n          @/*\\n\\tpublic /*@ pure
57   /*/ boolean contains(int elem) {/*...*/ throw new RuntimeException(\"Not yet
58   implemented\");}\n\\n\\t    private void provokeWarning() {\n          @\n          switch (1) {\n            case 1:\n              System.out.println(\"Hello World\");\n              case 2: //
59              Theres a fall-through warning around here\n              break;\n            }\n          }\\n\\t/*@ public normal_behavior\\n\\t @ requires contains(elem);\n          @ assignable
60   size, arr[*]; // allows arbitrary reordering of the array elements\\n          @ ensures
61   !contains(elem);\n          @ ensures (\\forall int e;\n          @\n          e != elem;\n          @
62   contains(e) <==> \\old(contains(e));)\n          @ ensures size ==
63   \\old(size) - 1;\n          @ also\\n          @ \\n          @ public normal_behavior\\n          @
64   requires !contains(elem);\n          @ assignable \\nothing;\n          @/*\\n\\tpublic void
65   remove(int elem) {/*...*/}\\n\\n\\t// we specify that the array is sorted afterwards and
66   that the set has not changed; the latter works in this case and is easier \\n\\t// than if
67   we would have to try to formalize permutation\\n\\t/*@ public normal_behavior\\n\\t @
68   assignable a[0..size - 1];\n          @ ensures\\n          @ (\\forall int i, j; i>=0 && i<j &&
69   j<size; arr[i]<arr[j]);\n          @ ensures (\\forall int e;\n          @
70   contains(e) <==> \\old(contains(e));)\n          @/*\\n\\tpublic void sort() { /* ... */
71   }\\n\\n\\t/*@ public normal_behavior\\n\\t @ requires size > 0;\n          @ assignable
72   \\nothing;\n          @ ensures (\\forall int i;\n          @\n          i>=0 && i<size;\n          @
73   \\result >= a[i]);\n          @ ensures (\\exists int i;\n          @
74   i>=0 && i<size;\n          @\n          \\result == a[i]);\n          @\\n          @ also\\n
75          @ \\n          @ public exceptional_behavior\\n          @ requires size == 0;\n          @ assignable
76          @\\nothing;\n          @ signals (RuntimeException) true;\n          @/*\\n\\tpublic int max()
77   {\n\\n\\t\\tthrow new RuntimeException(\"Not yet implemented.\");\n\\n\\t}\\n\\n"
78
79 responses:
80 '200':
81   description: OK. Successful static analysis. Returning results
82   schema:
83     type: array
84     items:
85       $ref: '#/definitions/Diagnostic'
86
87 /projects:
88   get:
89     tags:
90       - project-controller
91     summary: List projects stored on server
92     operationId: listProjectsUsingGET
93     produces:
94       - application/json; charset=UTF-8
95     responses:
96       '200':
97         description: OK. A list of project names
98         schema:
99           type: array
100          items:
101            type: string
102            example:
103              - My Project
104              - Key
105
106        deprecated: false
107
108 /projects/**:
109   get:
110     tags:
111       - project-controller
112     summary: Load content and name of specific file on the server
113     description: >-
114       Replace `***` with any file path, starting with the project name as root directory.
115
116       For example: `/projects/SimpleJML/src/examples/IntegerUtil.java`
117     operationId: openFileUsingGET

```

```

87      produces:
88          - '*/*'
89      responses:
90          '200':
91              description: OK
92              schema:
93                  $ref: '#/definitions/OpenedFileResponse'
94          '500':
95              description: Internal server error. The file could not be read, though it exists.
96              schema:
97                  type: string
98          '404':
99              description: Not Found. The requested file could not be found within the project.
100             schema:
101                 type: string
102             deprecated: false
103     '/projects/{projectname}':
104         get:
105             tags:
106                 - project-controller
107             summary: Load file tree of a specific project stored on the server
108             operationId: showProjectUsingGET
109             produces:
110                 - application/json; charset=UTF-8
111             parameters:
112                 - name: projectname
113                     in: path
114                     description: Name of the project, for which the file tree shall be retrieved.
115                     required: true
116                     type: string
117                     x-example: Hello World
118             responses:
119                 '200':
120                     description: >-
121                         OK. Tree file structure of a (java) project. Consists of files and folders.
122                         The name of the root folder is the same as the project name.
123                         (The data structure is recursive, which OpenAPI 2.0 descriptions can not
124                         express, therefore the specified model lacks the recursion.)
125                     schema:
126                         $ref: '#/definitions/FileOrFolderItem'
127                 '500':
128                     description: Internal Server Error. Occurs, if project does not exist.
129                     # TODO: This should be 400 instead.
130         delete:
131             tags:
132                 - project-controller
133             summary: Deletes a project
134             operationId: deleteProjectUsingDELETE
135             produces:
136                 - application/json; charset=UTF-8
137             parameters:
138                 - name: projectname
139                     in: path
140                     description: Name of the project which shall be deleted.
141                     required: true
142                     type: string
143                     x-example: Hello World
144             responses:
145                 '200':
146                     description: OK. Project got successfully deleted.
147                     schema:
148                         type: object
149                 '500':
150                     description: Internal Server Error. The project exists, but could not be deleted.
151                     schema:
152                         type: string

```

```

153     '404':
154         description: Not found. There exists no such project on the server.
155         schema:
156             type: string
157     '/projects/{projectname}/**':
158         post:
159             tags:
160                 - project-controller
161             summary: Update the contents of a file stored the server or rename it.
162             description: >-
163                 1. Allows to save a file, such that other clients may access it using the server API.
164
165                 2. Also allows to rename a file.
166
167
168             If the field `fileContent` is not set in the body, case 2 applies.
169
170             Otherwise, if `fileContent` is set, the content of the file
171             will be replaced with the contents of `fileContent`, such that all subsequent calls
172             to GET `/projects/<projectname>/<path-to-file>` will return the updated content.
173             In this case, the file name will not be changed, even if `fileName` is set, too.
174             It is assumed, that the file content is encoded in UTF-8.
175
176             Replace `**` with a path to a file within the specified project.
177
178             For example: `src/example/IntegerUtils.java`
179             operationId: updateFileUsingPOST
180             consumes:
181                 - application/json; charset=UTF-8
182             produces:
183                 - application/json; charset=UTF-8
184             parameters:
185                 - name: projectname
186                     in: path
187                     description: Name of the project of which we'd like to change the content of a file.
188                     required: true
189                     type: string
190                 - in: body
191                     name: updateData
192                     description: Name and new content
193                     required: true
194                     schema:
195                         $ref: '#/definitions/FileUpdateData'
196             responses:
197                 '200':
198                     description: >-
199                         OK. The change was successfully applied.
200
201                     The new project tree is returned, see `GET /projects/{projectname}`.
202
203                     If the file name has not been changed, the returned tree will also appear unchanged,
204                     given no other client changed the project structure in the meantime.
205                     schema:
206                         $ref: '#/definitions/FileOrFolderItem'
207                 '400':
208                     description: >-
209                         Bad request. The file does not exist, or the rename path is invalid.
210
211                         Can also be caused by an internal error while updating.
212                     schema:
213                         type: string
214                         # TODO: Throw internal server error (500) in that last case.
215             put:
216                 tags:
217                     - project-controller
218                     summary: Create a new file or folder on the server.

```

```

219     description: >-
220         Allows to create a file or folder on the server, such that other clients may access
221         it using the server API.
222
223         Replace `***` with a path within the specified project.
224
225         For example: `src/example/IntegerUtils.java`
226     operationId: createFileUsingPUT
227     consumes:
228         - application/json
229     produces:
230         - application/json;charset=UTF-8
231     parameters:
232         - name: projectName
233             in: path
234             description: Name of the project, within we want to create a new file/folder
235             required: true
236             type: string
237         - name: type
238             in: query
239             description: >-
240                 Indicates, whether to create a file or folder.
241
242                 Possible values: 'file', 'folder'
243             required: true
244             type: string
245     responses:
246         '200':
247             description: >-
248                 OK. The file/folder was successfully created.
249
250             The updated project tree is returned, see `GET /projects/{projectname}`.
251     schema:
252         $ref: '#/definitions/FileOrFolderItem'
253     '400':
254         description: >-
255             Bad request.
256             Sent in the following cases:
257
258             1. There already exists a file/folder at the given path.
259
260             2. The type parameter has an invalid value. It must equal either 'file' or 'folder'
261     schema:
262         type: string
263     delete:
264         tags:
265             - project-controller
266         summary: Delete a file or folder from a project
267         operationId: deleteFileUsingDELETE
268         produces:
269             - application/json;charset=UTF-8
270     parameters:
271         - name: projectName
272             in: path
273             description: Name of the project, where a file/folder shall be deleted
274             required: true
275             type: string
276     responses:
277         '200':
278             description: >-
279                 OK. The file/folder was successfully deleted.
280
281             The updated project tree is returned, see also `GET /projects/{projectname}`.
282     schema:
283         $ref: '#/definitions/FileOrFolderItem'
284     '404':

```

```

285     description: >-
286         Not found.
287         Returned, if the specified file/folder does not exist or the path is invalid.
288     schema:
289         type: string
290     '500':
291         description: >-
292             Internal server error.
293             The file/folder which shall be deleted does exist, but could not be deleted.
294         schema:
295             type: string
296         deprecated: false
297     '/proof/**/{className}.java':
298         get:
299             tags:
300                 - proof-controller
301             summary: Prove obligations within a Java file using KeY
302             description: >-
303                 This method will try to prove the specified proof obligations within the given file
304                 using KeY (https://www.key-project.org/)
305
306                 If no obligation indices are specified, the server will try to prove all obligations
307                 within the given file.
308             operationId: runProofUsingGET
309             produces:
310                 - application/json; charset=UTF-8
311             parameters:
312                 - name: className
313                     in: path
314                 description: >-
315                     Name of the class in which the proofs shall be run.
316             required: true
317             type: string
318             - name: obligationIdxs
319                 in: query
320                 description: >-
321                     Indices of the proof obligations, which shall be proven, counted from the beginning
322                     of the Java file.
323             required: false
324             type: array
325             items:
326                 type: integer
327                     format: int32
328             - name: macro
329                 in: query
330                 type: string
331                 description: >-
332                     Path to the macro file to use for the requested proof(s), including the project name. Can be
333                     ↵ omitted if the proof should be executed without a macro/ proof script.
334             required: false
335             responses:
336                 '200':
337                     description: >-
338                         OK. If an obligation could not be proven, the reasons are encoded within the
339                         returned structure.
340
341                         Beware of the example return value generated by Swagger, which can for example contain
342                         ↵ successful proofs in the failed list. This is because OpenAPI 2.0's way of specifying
343                         ↵ examples is very restricted.
344             schema:
345                 $ref: '#/definitions/ProofResult'
346     '/proof/**/{className}.java/obligation':
347         get:
348             tags:
349                 - proof-controller
350             summary: Lists proof obligation indices of saved proofs.

```

```

348     description: >-
349         Lists all obligation indices for the given file, for which there are saved proofs available on the
350         → server.
351
352         The indices are counted per file from top to bottom for each discovered proof obligation.
353
354         The returned indices can be used to retrieve saved proofs using `GET
355         → /proof/**/{className}.java/obligation/{obligation index}/**`.
356     operationId: listSavedObligationsUsingGET
357     produces:
358         - application/json; charset=UTF-8
359     parameters:
360         - name: className
361         in: path
362         description: Name of the class, for which we want to list obligation indices of saved proofs.
363         required: true
364         type: string
365     responses:
366         '200':
367             description: >-
368                 OK.
369
370                 This method will always return code 200.
371                 If there are no saved proofs, an empty list will be returned.
372             schema:
373                 type: array
374                 items:
375                     type: integer
376                     format: int32
377                     example: [0, 2, 3]
378     '/proof/**/{className}.java/obligation/{obligationIdx}/history':
379         get:
380             tags:
381                 - proof-controller
382             summary: List ids of proofs saved to history
383             description: >-
384                 The server can save proof results within a history, such that they can be retrieved and reviewed
385                 → at a
386                 later time.
387
388                 Each saved proof result is given a history id, which can be used to retrieve a result using `GET
389                 → /proof/**/{className}.java/obligation/{obligationIdx}/history/{history id}`.
390
391             The id also decodes the age of a saved result, more recent proof results get assigned bigger ids
392             → than their predecessors.
393         operationId: getHistoryItemsUsingGET
394         produces:
395             - application/json; charset=UTF-8
396         parameters:
397             - name: className
398             in: path
399             description: Name of the class, for which we want to retrieve proof history information.
400             required: true
401             type: string
402             - name: obligationIdx
403             in: path
404             description: >-
405                 A history is maintained for each proof obligation.
406
407                 This is the index of the obligation counted from the top of the file, for which proof result
408                 history information shall be retrieved.
409             required: true
410             type: integer
411             format: int32
412         responses:
413             '200':

```

```

409     description: >-
410     OK.
411
412     This method will always return code 200.
413     If there are no saved proofs, an empty list will be returned.
414   schema:
415     type: array
416     items:
417       type: integer
418       format: int32
419     example: [2, 20, 3]
420
421   post:
422     tags:
423       - proof-controller
424     summary: Add a proof result to the server history.
425     description: >-
426       The server can save proof results within a history for each proof obligation, for each file, such
427       ↳ that they can be retrieved and reviewed at a later time.
428
429       Using this route, a proof result can be added to the history.
430   operationId: addToHistoryUsingPOST
431   consumes:
432     - application/json
433   produces:
434     - application/json;charset=UTF-8
435   parameters:
436     - name: className
437       in: path
438       description: Name of the class, for which we want to save a proof result.
439       required: true
440     - name: obligationIdx
441       in: path
442       description: >-
443         Index of the proof obligation the saved proof belongs to,
444         counted from top to bottom in the given Java file.
445       required: true
446       type: integer
447       format: int32
448     - in: body
449       name: obligationResult
450       description: Proof result, which shall be saved.
451       required: true
452       schema:
453         $ref: '#/definitions/ObligationResult'
454   responses:
455     '200':
456       description: OK. Returns the unique history id of the saved proof.
457       schema:
458         type: integer
459         format: int32
460         example: 3
461   '/proof/**/{className}.java/obligation/{obligationIdx}/history/{historyIdx}':
462     get:
463       tags:
464         - proof-controller
465       summary: Retrieve a proof from the server history.
466       description: >-
467         The server can save proof results within a history for each proof obligation, for each file, such
468         ↳ that they can be retrieved and reviewed at a later time.
469
470       Using this route, a proof result can be retrieved from the history.
471   operationId: getHistoricProofUsingGET
472   produces:
473     - application/json;charset=UTF-8
474   parameters:

```

```

473     - name: className
474         in: path
475         description: Name of the class, for which we want to retrieve a proof result.
476         required: true
477         type: string
478     - name: obligationIdx
479         in: path
480         description: >-
481             Index of the proof obligation the proof which shall be retrieved belongs to,
482             counted from top to bottom in the given Java file.
483         required: true
484         type: integer
485         format: int32
486     - name: historyIdx
487         in: path
488         description: >-
489             Unique history id of the saved proof.
490             See `GET /proof/**/{className}.java/obligation/{obligationIdx}/history`
491         required: true
492         type: integer
493         format: int32
494     responses:
495     '200':
496         description: OK. Returns the saved proof result.
497         schema:
498             $ref: '#/definitions/ObligationResult'
499     '404':
500         description: Not found. There is no proof result saved for the given path and parameters.
501 delete:
502     tags:
503         - proof-controller
504     summary: Delete a proof result from the server history.
505     description: >-
506         The server can save proof results within a history for each proof obligation, for each file, such
507         ↳ that they can be retrieved and reviewed at a later time.
508
509         Using this route, a proof result can be deleted from the history.
510     operationId: deleteFromHistoryUsingDELETE
511     produces:
512         - application/json; charset=UTF-8
513     parameters:
514         - name: className
515             in: path
516             description: Name of the class, for which we want to delete a proof result from the history.
517             required: true
518             type: string
519         - name: historyIdx
520             in: path
521             description: >-
522                 Unique history id of the proof result which shall be deleted from the history.
523                 See `GET /proof/**/{className}.java/obligation/{obligationIdx}/history`
524             required: true
525             type: integer
526             format: int32
527         - name: obligationIdx
528             in: path
529             description: >-
530                 Index of the proof obligation the proof which shall be deleted belongs to,
531                 counted from top to bottom in the given Java file.
532             required: true
533             type: integer
534             format: int32
535     responses:
536     '200':
537         description: OK. Successfully deleted proof result from history.
538     '404':

```

```

538     description: >-
539         Not found.
540         The specified proof result can not be deleted from history, since there is no such
541         result saved.
542     '/proof/**/{className}.java/obligation/{obligationIdx}/last':
543         get:
544             tags:
545                 - proof-controller
546             summary: Retrieve the latest temporary proof result.
547             description: >-
548                 In addition to a permanent proof result history, the server can temporarily store the latest prove
549                 → result. It is lost as soon as the server is shut down.
550
551                 This feature is used by KollaborierbaR to always share the latest proof result by any developer
552                 → with everyone working on the same file.
553
554                 Using this route, this temporarily stored result can be retrieved.
555             operationId: getCurrentProofUsingGET
556             produces:
557                 - application/json;charset=UTF-8
558             parameters:
559                 - name: className
560                     in: path
561                     description: >-
562                         Name of the class, for which we want to retrieve the latest, temporary proof result.
563                         required: true
564                         type: string
565                 - name: obligationIdx
566                     in: path
567                     description: >-
568                         Index of the proof obligation for which the latest temporary results shall be retrieved,
569                         counted from top to bottom in the given Java file.
570                         required: true
571                         type: integer
572                         format: int32
573             responses:
574                 '200':
575                     description: OK. Returns the temporarily saved proof result.
576                     schema:
577                         $ref: '#/definitions/ObligationResult'
578                 '404':
579                     description: >-
580                         Not found.
581                         There has no temporary proof result been saved for the given path and parameters.
582             post:
583                 tags:
584                     - proof-controller
585                 summary: Save a temporary proof result.
586                 description: >-
587                     In addition to a permanent proof result history, the server can temporarily store the latest prove
588                     → result. It is lost as soon as the server is shut down.
589
590                     This feature is used by KollaborierbaR to always share the latest proof result by any developer
591                     → with everyone working on the same file.
592
593                     Using this route, such a temporarily result can be uploaded.
594             operationId: uploadCurrentObligationResultUsingPOST
595             consumes:
596                 - application/json
597             produces:
598                 - application/json;charset=UTF-8
599             parameters:
600                 - name: className
601                     in: path
602                     description: >-
603                         Name of the class, for which we want to save a temporary proof result.

```

```

600      required: true
601      type: string
602      - name: obligationIdx
603          in: path
604          description: >-
605              Index of the proof obligation to which the uploaded result belongs,
606              counted from top to bottom in the given Java file.
607      required: true
608      type: integer
609      format: int32
610      - in: body
611          name: obligationResult
612          description: proof result to be stored
613          required: true
614          schema:
615              $ref: '#/definitions/ObligationResult'
616      responses:
617          '200':
618              description: OK. The proof result has been temporarily stored.
619  definitions:
620      Diagnostic:
621          type: object
622          properties:
623              endCol:
624                  type: integer
625                  format: int64
626                  example: 42
627              endRow:
628                  type: integer
629                  format: int64
630                  example: 24
631              kind:
632                  type: string
633                  enum:
634                      - ERROR
635                      - WARNING
636                      - NOTE
637                      - NOT_SUPPORTED
638              message:
639                  type: string
640                  example: Syntax error
641              startCol:
642                  type: integer
643                  format: int64
644                  example: 0
645              startRow:
646                  type: integer
647                  format: int64
648                  example: 24
649              title: Diagnostic
650          example:
651              endCol: 42
652              endRow: 24
653              kind: ERROR
654              message: Syntax error
655              startCol: 0
656              startRow: 24
657  FileUpdateData:
658      type: object
659      properties:
660          fileContent:
661              type: string
662              example: public class Main {\n    public static void main(String[] args) {}}\n
663          fileName:
664              type: string
665              example: /projects/src/main/java/Main.java

```

```

666     title: FileUpdateData
667     FileOrFolderItem:
668     type: object
669     required:
670       - name
671       - type
672     properties:
673       contents:
674         type: array
675         items:
676           $ref: '#/definitions/FileOrFolderItem'
677         description: >-
678           Contents of the folder. This field is not set, if the item is a file, not a folder.
679       name:
680         type: string
681         example: Main.java
682       type:
683         type: string
684         enum:
685           - file
686           - folder
687     title: FileOrFolderItem
688     ObligationResult:
689     type: object
690     required:
691       - kind
692       - obligationIdx
693       - openGoals
694       - resultMsg
695       - targetName
696     properties:
697       id:
698         type: number
699         description: >-
700           Unique id used to identify obligation results in other routes
701       kind:
702         type: string
703         enum:
704           - success
705           - failure
706           - error
707       obligationIdx:
708         type: integer
709         format: int32
710         example: 0
711         description: >-
712           Obligation index this proof belongs to.
713           It is counted from top to bottom in the corresponding source file.
714       openGoals:
715         type: array
716         items:
717           $ref: '#/definitions/OpenGoalInfo'
718       proofTree:
719         $ref: '#/definitions/ProofNode'
720       resultMsg:
721         type: string
722         example: "Contract 'JML normal_behavior operation contract 0' of IntegerUtil2::add is verified."
723         description: Human readable statement about the result state of the proof.
724       targetName:
725         type: string
726         example: "IntegerUtil2::add"
727         description: Target subject of the proof, for example a method name.
728     title: ObligationResult
729     OpenGoalInfo:
730     type: object
731     required:

```

```

732     - formula
733     - id
734     - sequent
735   properties:
736     formula:
737       type: string
738       example: "wellFormed(heap), measuredByEmpty ==> y = 0\n"
739       description: Created by calling toString on KeY goals, may not be equal to `sequent`, but
740         ↪ oftentimes is.
741     id:
742       type: integer
743       format: int64
744       example: 64
745     sequent:
746       type: string
747       example: "wellFormed(heap), measuredByEmpty ==> y = 0\n"
748       description: Created using the quickPrintSequent function of KeY's LogicPrinter. May not be
749         ↪ equal to 'formula'.
750   title: OpenGoalInfo
751   OpenedFileResponse:
752     type: object
753     properties:
754       fileName:
755         type: string
756         example: 'IntegerUtil.java'
757       fileText:
758         type: string
759         example: "public class IntegerUtil {\n/*@ normal_behavior\n    @ ensures \\\result == x + y;\n    @\n    public static int add(int x, int y) {\n        return x + y;\n    }\n    */@\n    @ normal_behavior\n    @ requires (x != y);\n    @ ensures \\\result == x - y;\n    @\n    @\n    @ exceptionalBehaviour\n    @ requires (x == y);\n    @ signals (RuntimeException) true;\n    @\n    @\n    public static int sub(int x, int y) {\n        if (x == y) {throw new\n            RuntimeException();}\n        return x + y;\n    }\n}""
760   ProofNode:
761     type: object
762     properties:
763       children:
764         type: array
765         items:
766           $ref: '#/definitions/ProofNode'
767       kind:
768         type: string
769         enum:
770           - ClosedProofTree
771           - OneStepSimplification
772           - OpenProofTree
773           - BranchNode
774           - OpenGoal
775           - ClosedGoal
776           - InteractiveGoal
777           - LinkedGoal
778           - DefaultNode
779       oneStepId:
780         type: integer
781         format: int32
782         example: 0
783         description: >-
784           Unique id of this node relative to all its siblings,
785           if it is a child of a One Step Simplification node.
786
787           Otherwise, it always equals 0.
788
789           Together with 'serialNr', this field forms a unique id within the whole proof tree.
790   sequent:
791     type: string
792     example: " \n==>\n wellFormed(heap)\n & ... "

```

```

791     serialNr:
792         type: integer
793         format: int32
794         example: 0
795         description: >-
796             Unique id of this node within the proof tree, if its parent is not a One Step Simplification.
797             If its parent is a One Step simplification, it forms a unique key together with `oneStepId`.
798     text:
799         type: string
800         example: "IntegerUtil2[IntegerUtil2::add(int,int)].JML normal_behavior operation contract.0"
801     title: ProofNode
802 ProofResult:
803     type: object
804     properties:
805         failed:
806             type: array
807             items:
808                 $ref: '#/definitions/ObligationResult'
809                 description: Proofs, which could not be closed.
810         succeeded:
811             type: array
812             items:
813                 $ref: '#/definitions/ObligationResult'
814                 description: Successful proofs.
815     title: ProofResult

```

Server API Tests



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Contents

1 Server API Tests	2
1.1 Tests für LinterController	2
1.2 Tests für ProjectController	3
1.3 Tests für ProofController	11

1 Server API Tests

Beim Testen haben wir darauf geachtet, die Server API auf alle Fälle zu testen, welche durch die Spezifikation definiert werden. Es folgt eine Auflistung aller Tests, geordnet nach den dazugehörigen Routen.

1.1 Tests für LinterController

Das Testen der Server API führen wir mittels des Karate¹ Frameworks durch. Karate erlaubt es die Testfälle in einer *Domain Specific Language* zu formulieren und vom Java Code aus aufzurufen.

Die Testfälle können wie folgt in *.feature* Dateien formuliert werden:

```
1  Feature: Request linting of java code from the server
2
3  Scenario: Lint code with an error and get the correct line and column (starting with 0)
4
5  Given url 'http://localhost:9000/lint?name=Test.java'
6  And request "public class Test{\npublic static void main(String[] args){\nsyntaxerror\n}\n}"
7  When method post
8  Then status 200
9  And match response contains
10   """
11 {
12   message: "#string",
13   endCol:11,
14   endRow:2,
15   kind:"ERROR",
16   startCol:0,
17   startRow:2
18 }
19 """
20
21 Scenario: Lint code with an unsupported feature and get the correct line and column (starting with 0)
22
23 Given url 'http://localhost:9000/lint?name=Test.java'
24 And request "public class Test{\nimport java.util.function.Consumer\npublic static void main(String[]
25   → args){\nConsumer<String> test = s -> {};\n}\n}"
26 When method post
27 Then status 200
28 And match response contains
29 """
30 {
31   message: "#string",
32   endCol:31,
33   endRow:3,
34   kind:"NOT_SUPPORTED",
35   startCol:24,
36   startRow:3
37 }
38 """
```

Zusätzlich ist es notwendig *.feature* Dateien in den Javacode einzubinden:

¹ <https://github.com/intuit/karate>

```

1 package projectmanagement;
2
3 import com.intuit.karate.KarateOptions;
4 import com.intuit.karate.junit4.Karate;
5 import org.junit.runner.RunWith;
6
7 @RunWith(Karate.class)
8 @KarateOptions(features = "classpath:projectcontroller/LintFile.feature")
9 public class LintFile {}
```

1.2 Tests für ProjectController

Vor und nach jedem Test der `/project/**` API wird ein Setup und ein Cleanup Schritt durchgeführt. Dies dient dazu, die Datenbank mit den richtigen Daten vorzubereiten und danach Datenüberbleibsel zu entfernen. Dies wurde mit den folgenden zwei Anfragen realisiert:

```

1 # This feature file isn't a test by it self. It is supposed
2 # to be called (with call read ('HttpApiTestSetup.feature')) before
3 # testing other Features/Scenarios
4 Feature: Setup for Http Api Testing
5 Scenario: Setup for Http Api Testing
6
7 #Create testProject1
8 Given url 'http://localhost:9000/projects/testProject1?type=folder'
9 And request {}
10 When method put
11
12 #Create testProject2
13 Given url 'http://localhost:9000/projects/testProject2?type=folder'
14 And request {}
15 When method put
16
17 #Create testProject3
18 Given url 'http://localhost:9000/projects/testProject3?type=folder'
19 And request {}
20 When method put
21
22 #Create testProject1/testFolder1
23 Given url 'http://localhost:9000/projects/testProject1/testFolder1?type=folder'
24 And request {}
25 When method put
26
27 #Create testProject1/testFolder1/testFile1
28 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1?type=file'
29 And request {}
30 When method put
31
32 #Create testProject1/testFolder1/testFile2
33 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile2?type=file'
34 And request {}
35 When method put
36
37 #Create testProject1/testFolder1/testFile3
38 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile3?type=file'
39 And request {}
```

```

40 When method put
41
42 #Create testProject1/testFolder2/
43 Given url 'http://localhost:9000/projects/testProject1/testFolder2?type=folder'
44 And request {}
45 When method put
46
47 #Create testProject1/testFolder2/testSubFolder1
48 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder1?type=folder'
49 And request {}
50 When method put
51
52 #Create testProject1/testFolder2/testSubFolder2
53 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder2?type=folder'
54 And request {}
55 When method put
56
57 #Create testProject1/testFolder2/testSubFolder3
58 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder3?type=folder'
59 And request {}
60 When method put
61
62 #Create testProject1/testFolder2/testSubFolder3/testSubFile1
63 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder3/testSubFile1?type=file'
64 And request {}
65 When method put
66
67 # Set testProject1/testFolder2/testSubFolder3/testSubFile1 fileContent
68 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder3/testSubFile1'
69 And request {"fileContent" : "Test1"}
70 When method post

```

```

1 # This feature file isn't a test by it self. It is supposed
2 # to be called (with call read ('HttpApiTestCleanUp.feature')) after
3 # testing other Features/Scenarios
4 Feature: CleanUp for Http Api Testing
5 Scenario: CleanUp for Http Api Testing
6
7 #CleanUp all projects created with HttpApiTestSetup.feature
8 Given url 'http://localhost:9000/projects/testProject1'
9 And request {}
10 When method delete
11
12 Given url 'http://localhost:9000/projects/testProject2'
13 And request {}
14 When method delete
15
16 Given url 'http://localhost:9000/projects/testProject3'
17 And request {}
18 When method delete

```

Im folgenden eine Liste aller Tests an den *ProjectController*:

```

1 Feature: Creation of files/folders/projects
2
3
4
5 # This is just cleanup! If a previous test failed,
6 # there maybe already is a testProject4, which would
7 # prohibit the test from working
8 Scenario: Cleanup
9
10 # Delete testProject4
11 Given url 'http://localhost:9000/projects/testProject4'
12 And request {}
13 When method delete
14
15
16 # Start the real testing
17 Scenario: Create project
18
19
20 # Create testProject4
21 Given url 'http://localhost:9000/projects/testProject4?type=folder'
22 And request {}
23 When method put
24 Then status 200
25
26 # Check whether projects/testProject4 exists
27 Given url 'http://localhost:9000/projects/'
28 When method get
29 And request {}
30 Then status 200
31 And match response contains ["testProject4"];
32
33
34 Scenario: Create folder within newly created testProject4
35
36
37 Given url 'http://localhost:9000/projects/testProject4/testSubFolder1?type=folder'
38 And request {}
39 When method put
40 Then status 200
41
42 # Check whether testProject4/testSubFolder1 exists
43 Given url 'http://localhost:9000/projects/testProject4'
44 And request {}
45 When method get
46 Then status 200
47 And match response.contents contains {name : "testSubFolder1", type : "folder", contents: [] }
48
49
50 Scenario: Create file within newly created testProject4/testSubFolder1/
51
52
53 Given url 'http://localhost:9000/projects/testProject4/testSubFolder1/testSubFile1?type=file'
54 And request {}
55 When method put
56 Then status 200
57
58 # Check whether testProject4/testSubFolder1/testSubFile1 exists
59 Given url 'http://localhost:9000/projects/testProject4'
60 And request {}
61 When method get
62 Then status 200
63 Then def testSubFolder1Content =
64 """
65 [
66     {

```

```

67             name : "testSubFile1",
68             type : "file"
69         }
70     ]
71 """
72 And match response.contents contains {name : "testSubFolder1", type : "folder", contents:
73   ↳ '#(^testSubFolder1Content)' }
74
75 Scenario: Try to create testProject4/testSubFolder1/testSubFile1 again (supposed to throw error)
76
77
78 Given url 'http://localhost:9000/projects/testProject4/testSubFolder1/testSubFile1?type=file'
79 And request {}
80 When method put
81 Then status 400
82
83
84 Scenario: Try to create file with invalid type, for example ?type = SomethingWrong (supposed th throw
85   ↳ error)
86
87 Given url 'http://localhost:9000/projects/testProject4/testSubFolder1/testSubFile2?type=SomethingWrong'
88 And request {}
89 When method put
90 Then status 400
91
92 # This is cleanup!
93 # Delete testProject4
94 Given url 'http://localhost:9000/projects/testProject4'
95 And request {}
96 When method delete

```

```

1 Feature: Deletion of projects
2
3 Background:
4 # Cleanup possible old data (from past tests)
5 * call read('HttpApiTestCleanUp.feature')
6
7 # Setup /projects
8 * call read('HttpApiTestSetup.feature')
9
10
11 Scenario: Succesfully delete a project
12
13 # delete testProject1
14 Given url 'http://localhost:9000/projects/testProject1'
15 And request {}
16 When method delete
17 Then status 200
18
19 # check that testProject1 isn't in the list of all projects anymore
20 Given url 'http://localhost:9000/projects'
21 And request {}
22 When method get
23 Then status 200
24 And match response !contains ["testProject1"]
25
26
27 Scenario: Unsuccesfully delete a project
28

```

```

29 # try to delete project which doesn't exists
30 Given url 'http://localhost:9000/projects/testProject10'
31 And request {}
32 When method delete
33 Then status 404
34
35
36 # Cleanup possible old data created with setup
37 * call read('HttpApiTestCleanUp.feature')

```

```

1 Feature: Get list of projects (GET /projects)
2 Scenario: Get list of projects (GET /projects)
3
4 # Cleanup possible old data (from past tests)
5 * call read('HttpApiTestCleanUp.feature')
6
7 # Setup /projects
8 * call read('HttpApiTestSetup.feature')
9
10 Given url 'http://localhost:9000/projects'
11 When method get
12 Then status 200
13 And match response contains ["testProject1", "testProject2", "testProject3"]
14
15 # Cleanup all data created with setup
16 * call read('HttpApiTestCleanUp.feature')

```

```

1 Feature: Open File with fileContent und fileName (GET /projects/)
2
3
4 Background:
5 # Cleanup possible old data (from past tests)
6 * call read('HttpApiTestCleanUp.feature')
7
8 # Setup /projects
9 * call read('HttpApiTestSetup.feature')
10
11 Scenario: Request a file which is present on the server
12
13
14 # Load testProject1/testFolder2/testSubFolder3/testSubFile1
15 Given url 'http://localhost:9000/projects/testProject1/testFolder2/testSubFolder3/testSubFile1'
16 When method get
17 And request {}
18 Then status 200
19 And match response contains
20 """
21 {"fileName" : "testSubFile1"}
22 """
23 And match response contains
24 """
25 {"fileText" : "Test1"}
26 """

```

```

27
28 Scenario: Request a file which isn't present on the server
29
30 Given url 'http://localhost:9000/projects/testProject1/someFileWhichDontExist'
31 When method get
32 And request {}
33 Then status 404
34
35
36 # Cleanup all data created with setup
37 * call read('HttpApiTestCleanUp.feature')

```

```

1 Feature: Update file (fileText and fileName) (POST /projects/)
2
3
4 Background:
5
6 # Cleanup possible old data (from past tests)
7 * call read('HttpApiTestCleanUp.feature')
8
9 # Setup /projects
10 * call read('HttpApiTestSetup.feature')
11
12 Scenario: Update fileContent of testProject1/testFolder1/testFile1
13
14 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1'
15 And request {"fileContent" : "Test2"}
16 When method post
17 Then status 200
18
19 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1'
20 And request {}
21 When method get
22 Then status 200
23 And match response contains
24 """
25 {"fileText" : "Test2"}
26 """
27
28 Scenario: Update fileName of testProject1/testFolder1/testFile1
29
30 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1'
31 And request {"fileName" : "/projects/testProject1/testFolder1/testFile1Rename"}
32 When method post
33 Then status 200
34
35 # Check whether /projects/testProject1/testFolder1/testFile1Rename exists
36 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1Rename'
37 And request {}
38 When method get
39 Then status 200
40
41 Scenario: Update file which dont exists (supposed to throw error)
42
43 Given url 'http://localhost:9000/projects/testProject1/fileWhichDontExists'
44 And request {"fileName" : "/projects/testProject1/tryToRename"}
45 When method post
46 Then status 400
47
48

```

```
49  # Cleanup possible old data (from past tests)
50  * call read('HttpApiTestCleanUp.feature')
```

```
1  Feature: Deletion of files and folders
2
3  Background:
4  # Cleanup possible old data (from past tests)
5  * call read('HttpApiTestCleanUp.feature')
6
7  # Setup /projects
8  * call read('HttpApiTestSetup.feature')
9
10
11 Scenario: Delete a folder from testProject1
12
13 # Delete testProject1/testFolder1
14 Given url 'http://localhost:9000/projects/testProject1/testFolder1'
15 And request {}
16 When method delete
17 Then status 200
18
19 # Check that testProject1/testFolder1 doesn't exists anymore
20 Given url 'http://localhost:9000/projects/testProject1'
21 And request {}
22 When method get
23 Then status 200
24 And match response.contents !contains {name: "testFolder1", type : "folder", content : '#ignore'}
25
26
27 Scenario: Delete a file from testProject1/testFolder1
28
29
30 Given url 'http://localhost:9000/projects/testProject1/testFolder1/testFile1'
31 And request {}
32 When method delete
33 Then status 200
34
35 # Check that testProject1/testFolder1/testFile1 doesn't exists anymore
36 Given url 'http://localhost:9000/projects/testProject1'
37 And request {}
38 When method get
39 Then status 200
40 Then def testFolder1Contents =
41 """
42 [
43     {
44         name : "testFile1",
45         type : "file"
46     }
47 ]
48 """
49 Then def projectContents =
50 """
51     {
52         name : "testFolder1",
53         type : "folder",
54         contents : '#(!^testFolder1Contents)'
55     }
56 """
57 And match response contains
58 """
```

```

59      {
60          name : "testProject1",
61          type : "folder",
62          contents : '#(<projectContents>)'
63      }
64      """
65
66 Scenario: Delete a folder from testProject1 which doesn't exist (supposed to throw error)
67
68 Given url 'http://localhost:9000/projects/testProject1/FolderDontExist'
69 And request {}
70 When method delete
71 Then status 404
72
73
74 * call read('HttpApiTestCleanUp.feature')

```

```

1 Feature: Get the structure of a specific project (GET /projects/{projectname})
2 Scenario: Get the structure of a specific project (GET /projects/{projectname})
3
4 # Cleanup possible old data (from past tests)
5 * call read('HttpApiTestCleanUp.feature')
6
7 # Setup /projects
8 * call read('HttpApiTestSetup.feature')
9
10 Given url 'http://localhost:9000/projects/testProject1'
11 And request {}
12 When method get
13 Then def folder1Contents =
14 """
15 [
16     {
17         name: "testFile1",
18         type: "file"
19     },
20     {
21         name: "testFile2",
22         type: "file"
23     },
24     {
25         name: "testFile3",
26         type: "file"
27     }
28 ]
29 """
30 Then def folder2Contents =
31 """
32 [
33     {
34         name: "testSubFolder1",
35         type: "folder",
36         contents: []
37     },
38     {
39         name: "testSubFolder2",
40         type: "folder",
41         contents: []
42     },
43     {
44         name: "testSubFolder3",

```

```

45     type: "folder",
46     contents: [
47       {
48         name: "testSubFile1",
49         type: "file"
50       }
51     ],
52   },
53 ]
54 """
55 Then def projectContents =
56 """
57 [
58   {
59     name: "testFolder2",
60     type: "folder",
61     contents: #(^folder2Contents)
62   },
63   {
64     name: "testFolder1",
65     type: "folder",
66     contents: #(^folder1Contents)
67   },
68 ]
69 """
70 And match response contains
71 """
72 {
73   name: "testProject1",
74   type: "folder",
75   contents: '#(^projectContents)'
76 }
77 """
78
79 # Cleanup all data created with setup
80 * call read('HttpApiTestCleanUp.feature')

```

1.3 Tests für ProofController

```

1 Feature: List all the obligation indices from the given file, which have a saved proof
2
3
4 Scenario: List all the obligation indices from the given file, which have a saved proof
5
6
7 # First of, we have to save some proof to obligations
8
9 # Save proof to obligation 0
10 Given url 'http://localhost:9000/proof/testProject1/testListObligations.java/obligation/0/history'
11 And request
12 """
13 {
14   "kind" : "success",
15   "obligationIdx" : 0,
16   "openGoals" : [],
17   "resultMsg" : "Message generated by test",
18   "targetName" : "test1"
19 }
20 """
21 When method post

```

```

22 Then status 200
23
24 # Save proof to obligation 1
25 Given url 'http://localhost:9000/proof/testProject1/testListObligations.java/obligation/1/history'
26 And request
27 """
28 {
29     "kind" : "success",
30     "obligationIdx" : 0,
31     "openGoals" : [],
32     "resultMsg" : "Message generated by test",
33     "targetName" : "test1"
34 }
35 """
36 When method post
37 Then status 200
38
39 # Save proof to obligation 2
40 Given url 'http://localhost:9000/proof/testProject1/testListObligations.java/obligation/2/history'
41 And request
42 """
43 {
44     "kind" : "success",
45     "obligationIdx" : 0,
46     "openGoals" : [],
47     "resultMsg" : "Message generated by test",
48     "targetName" : "test1"
49 }
50 """
51 When method post
52 Then status 200
53
54 # Now the actual test starts, check whether the listObligations endpoint shows all the
55 # newly added proofs
56 Given url 'http://localhost:9000/proof/testProject1/testListObligations.java/obligation'
57 And request {}
58 When method get
59 Then match response contains [0, 1, 2]

```

```

1 Feature: Proving method contracts of files
2
3 Scenario: Setup
4 * call read('SetupKeyProject.feature')
5
6 Scenario: Successful proof with index
7 # Request proof for index 0
8 Given url 'http://localhost:9000/proof/testProject5/Test.java?obligationIdxs=0'
9 And request {}
10 When method get
11 Then call read('DefineTypes.feature')
12 And status 200
13 And match response ==
14 """
15 {
16     failed:[],
17     succeeded:[
18         {
19             "id": null,
20             "kind": "success",
21             "obligationIdx": 0,
22             "openGoals": []
23     }
24 }

```

```

23     "proofTree": "#(proofTreeSchema)",
24     "resultMsg": "#string",
25     "targetName": "#string"
26   }
27 ]
28 }
"""
29
30
31 Scenario: Prove all contracts with success and failure
32
33 # Request proofs for all indices
34 Given url 'http://localhost:9000/proof/testProject5/Test.java'
35 And request {}
36 When method get
37 Then call read('DefineTypes.feature')
38 And status 200
39 And match response ==
40 """
41 {
42   failed:[
43     {
44       "id": null,
45       "kind": "failure",
46       "obligationIdx": 1,
47       "openGoals": "#[] openGoalSchema",
48       "proofTree": "#(proofTreeSchema)",
49       "resultMsg": "#string",
50       "targetName": "#string"
51     }
52   ],
53   succeeded:[
54     {
55       "id": null,
56       "kind": "success",
57       "obligationIdx": 0,
58       "openGoals": [],
59       "proofTree": "#(proofTreeSchema)",
60       "resultMsg": "#string",
61       "targetName": "#string"
62     }
63   ]
64 }
"""
65
66 Scenario: Proving a file with syntax errors leads to a single error
67
68 # Insert incorrect java code into the test file
69 Given url 'http://localhost:9000/projects/testProject5/Test.java'
70 And request { fileContent: "mlem" }
71 When method post
72 Then status 200
73
74 # Request proofs for all contracts of the file
75 Given url 'http://localhost:9000/proof/testProject5/Test.java'
76 And request {}
77 When method get
78 Then call read('DefineTypes.feature')
79 And status 200
80 And match response ==
81 """
82 {
83   succeeded: [],
84   failed: [],
85 }
"""
86
87 Scenario: Cleanup

```

```

89
90 Given url 'http://localhost:9000/projects/testProject5'
91 And request {}
92 When method delete
93 Then status 200

```

```

1 Feature: Save and retrieve the last proof
2
3
4 #Scenario: Save the last proof
5 Scenario: Check whether the last proof (of the same obligation) is the one we just saved
6
7
8 Given url 'http://localhost:9000/proof/testProject1/SomeTest.java/obligation/1/last'
9 And request
10 """
11 {
12     "kind" : "success",
13     "obligationIdx" : 0,
14     "openGoals" : [],
15     "resultMsg" : "Message generated by test",
16     "targetName" : "test1"
17 }
18 """
19 When method post
20 Then status 200
21 Then def historyId = Number(response)
22
23
24
25 Given url 'http://localhost:9000/proof/testProject1/SomeTest.java/obligation/1/last'
26 And request {}
27 When method get
28 Then status 200
29 And match response contains
30 """
31 {
32     "kind" : "success",
33     "obligationIdx" : 0,
34     "openGoals" : [],
35     "resultMsg" : "Message generated by test",
36     "targetName" : "test1"
37 }
38 """
39
40
41 Scenario: Try to retrieve a proof which doesn't exist (supposed to throw error)
42
43
44 Given url 'http://localhost:9000/proof/testProject1/SomeTest.java/obligation/2/last'
45 And request {}
46 When method get
47 Then status 404

```

```

1 Feature: Setup a provable file
2
3 Scenario: Create file and project
4 # Create testProject5
5 Given url 'http://localhost:9000/projects/testProject5?type=folder'
6 And request {}
7 When method put
8 Then status 200
9
10 # Create Test.java
11 Given url 'http://localhost:9000/projects/testProject5/Test.java?type=file'
12 And request {}
13 When method put
14 Then status 200
15
16 # Insert java and jml into the file
17 Given url 'http://localhost:9000/projects/testProject5/Test.java'
18 And request { fileContent: "public class Test {\n/*@ normal_behavior\n @ ensures \\\result == x + y;\n→ */\npublic static int add(int x, int y) {\n    return x + y;\n}\n/*@ normal_behavior\n @ ensures\n→ \\\result == x + y;\n */\npublic static int sub(int x, int y) {\n    return x - y;\n}\n}" }
19 When method post
20 Then status 200

```

```

1 Feature: Delete proof results from history
2
3
4 Scenario: Successfully delete a proof from the history (proof has to exist)
5
6 # First we have to add a proof to the history
7 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/1/history'
8 And request {}
9 """
10 {
11     "kind" : "success",
12     "obligationIdx" : 0,
13     "openGoals" : [],
14     "resultMsg" : "Message generated by test",
15     "targetName" : "DeleteTest"
16 }
17 """
18 When method post
19 Then status 200
20 Then def historyId = response
21
22 # Now we have to check that the saved proof is present in the history
23 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/1/history/'
24 And request {}
25 When method get
26 Then status 200
27 And match response contains Number(historyId)
28
29
30 # Now delete the added proof from the history
31 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/1/history/' + historyId
32 And request {}
33 When method delete
34 Then status 200

```

```

35
36
37 # Now check that the delete proof isn't present in the history anymore
38 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/1/history/'
39 And request {}
40 When method get
41 Then status 200
42 And match response !contains Number(historyId)
43
44
45 Scenario: Try to delete a proof from the history which doesn't exist (supposed to throw an error)
46
47 Given url 'http://localhost:9000/proof/testProjectWrong/testWrong.java/obligation/3/history/' + 66
48 And request {}
49 When method delete
50 Then status 404

```

```

1 Feature: List all proofIds which belong to a proof obligation
2
3
4 Scenario: List all proofIds which belong to a proof obligation
5
6 # First of, we have to save some proof to obligations
7
8 # Save proof1 to obligation 0
9 Given url 'http://localhost:9000/proof/testProject1/testListProofIds.java/obligation/0/history'
10 And request
11 """
12 {
13     "kind" : "success",
14     "obligationIdx" : 0,
15     "openGoals" : [],
16     "resultMsg" : "Message generated by test",
17     "targetName" : "test1"
18 }
19 """
20 When method post
21 Then status 200
22 Then def historyId1 = Number(response)
23
24
25 # Save proof2 to obligation 0
26 Given url 'http://localhost:9000/proof/testProject1/testListProofIds.java/obligation/0/history'
27 And request
28 """
29 {
30     "kind" : "success",
31     "obligationIdx" : 0,
32     "openGoals" : [],
33     "resultMsg" : "Message generated by test",
34     "targetName" : "test1"
35 }
36 """
37 When method post
38 Then status 200
39 Then def historyId2 = Number(response)
40
41 # Save proof3 to obligation 0
42 Given url 'http://localhost:9000/proof/testProject1/testListProofIds.java/obligation/0/history'
43 And request
44 """

```

```

45      {
46          "kind" : "success",
47          "obligationIdx" : 0,
48          "openGoals" : [],
49          "resultMsg" : "Message generated by test",
50          "targetName" : "test1"
51      }
52      """
53 When method post
54 Then status 200
55 Then def historyId3 = Number(response)
56 Then print 'got history id: ', historyId3
57
58 # Now check that the list of proofIds which belong to obligation 0 is correct
59
60 Given url 'http://localhost:9000/proof/testProject1/testListProofIds.java/obligation/0/history'
61 And request {}
62 When method get
63 Then status 200
64 And match response contains ['#(historyId1)', '#(historyId2)', '#(historyId3)']

```

```

1 Feature: Prove Contract using a macro file
2
3 Scenario: Setup
4 * call read('SetupKeyProject.feature')
5
6 # Create a macro file in the project
7 Given url 'http://localhost:9000/projects/testProject5/Test.script?type=file'
8 And request {}
9 When method put
10 Then status 200
11
12 # Insert rubbish into the macro file to generate an error
13 Given url 'http://localhost:9000/projects/testProject5/Test.script'
14 And request { fileContent: "mulm" }
15 When method post
16 Then status 200
17
18
19 # Request proof for index 0 and expect an error
20 Given url 'http://localhost:9000/proof/testProject5/Test.java?obligationIdxs=0&macro=/Test.script'
21 And request {}
22 When method get
23 Then status 200
24 And call read('DefineTypes.feature')
25 And match response ==
26 """
27 {
28     succeeded: [],
29     failed: []
30 }
31 """
32
33 # Request proofs for all obligations and expect errors
34 Given url 'http://localhost:9000/proof/testProject5/Test.java?macro=/Test.script'
35 And request {}
36 When method get
37 Then status 200
38 And call read('DefineTypes.feature')
39 And match response ==
40 """

```

```

41  {
42      succeeded: [],
43      failed: [],
44  }
45 """
46
47 Scenario: Cleanup
48
49 Given url 'http://localhost:9000/projects/testProject5'
50 And request {}
51 When method delete
52 Then status 200

```

```

1 Feature: Add proof results to the history and retrieve it
2
3
4 Scenario: Add proof results to the history
5
6
7 # Add a proof result with the id 0 to the history of test.java
8 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/0/history'
9 And request {}
10 """
11 {
12     "kind" : "success",
13     "obligationIdx" : 0,
14     "openGoals" : [],
15     "resultMsg" : "Message generated by test",
16     "targetName" : "test1"
17 }
18 """
19 When method post
20 Then status 200
21 Then def historyId = Number(response)
22
23
24 # Now check that the proof result was saved in the history
25 Given url 'http://localhost:9000/proof/testProject1/test.java/obligation/0/history/' + historyId
26 And request {}
27 When method get
28 Then status 200
29 And match response contains
30 """
31 {
32     "id": "#(historyId)",
33     "kind" : "success",
34     "obligationIdx" : 0,
35     "openGoals" : [],
36     "resultMsg" : "Message generated by test",
37     "targetName" : "test1"
38 }
39 """

```

Ausgefüllte Fragebögen

Nutzerstudie: Ausgefüllte Evaluationsbögen

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

eventuell Icons hinzufügen (grünes + für create rotes X für delete) um Auswahlmöglichkeiten visuell aufzubessern

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Bitte Symbole

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Strg+Z bitte nur per Benutzer

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Ich habe zweimal anstelle eines Ordners eine Datei erstellt (oder umgekehrt), bitte Symbole im Kontextmenu.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Optionen zum Erstellen sollten über denen zum Löschen erscheinen.

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Bitte Sortierung (alphabetisch) von Dateien / Symbolen.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Ich habe den Kreis übersehen. --> Evtl. Rechtsklick auf Vertrag/Methode mit Kontextmenü start proof?

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Rückgängig macht Aktionen anderer Benutzer rückgängig.

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Man kann folder nur erstellen, wenn man einen anderen folder ausgewählt hat, nicht aber, wenn man ins "Leere" z.B. unter bestehende folder klickt. Verschieben von Dateien in andere Ordner ist nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

EinfachKlick für Ordner und Doppelklick für Files ist verwirrend. Gefühlt ist EinfachKlick intuitiver.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Die Punkte an der Seite neben den Methoden sind gut :) Bei den Rückmeldungen zu den Beweisen ließ sich nicht weiterscrollen (horizontal). Sonst wäre vielleicht eine Konsole unten gut, wo man durch einen Klick auf eine Rückmeldung die gesamte Meldung sehen kann (alternativ zum seitlichen Scrollen).

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Der Cursor war verbuggt, da er mit dem von meinem Teampartner identisch war.
Nach kurzem hin und her klicken war der Fehler weg.

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Ordner schließen sich ohne erkennbaren Grund.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Projekt schließt sich manchmal, beim gleichzeitigen löschen von Ordnern.

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Projekt konnte bei plötzlicher Schließung nur durch einen Neustart erneut geöffnet werden.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Der Name des Projektes konnte nach der Erstellung nicht mehr geändert werden.

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Ordner / Dateien sollten von überall aus erstellbar sein, nicht nur durch einen Rechtsklick auf den entsprechenden Überordner.

Ordner / Dateien sollten nach der Erstellung verschiebbar sein.

Ordner / Dateien / Projekte im Projekt-Auswahl-Menü sind nicht alphabetisch sortiert (bitte noch hinzufügen). Des weiteren sollten die Dateien / Ordner in der linken Leiste sortiert werden, am besten als erstes alle Ordner und darunter alle Dateien. Das erleichtert die Navigation und Dateifindung.

Ordner werden mit einem Klick geöffnet, Dateien aber mit zwei Klicks. Bitte auf eins der beiden einigen.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Siehe "Gab es beim Erstellen von Dateien und Ordnern Probleme?". Die Projekte sollten der Übersicht halber alphabetisch sortiert werden. Dies erleichtert auch die Projektfindung.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Das y in "Key" oben rechts bitte groß schreiben, also "KeY" als Name wählen.
Der Output des Beweises ist nicht skrollbar. Da der Output nur in einer Zeile angezeigt wird sollte es entweder möglich sein horizontal zu scrollen oder es sollten automatische Zeilenumbrüche eingefügt werden.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese lesen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Hinweis zur Dateibearbeitung:

Es gibt kein Autoscrolling.

Bei der Erstellung einer geöffneten Klammer wird direkt die geschlossene Klammer dazu erstellt, was gut ist. Allerdings wird beim drücken der geschlossenen Klammer nicht die erstelle geschlossene Klammer übersprungen ('|' stellt den Cursor dar):

$a + (b + c |) \rightarrow a + (b + c) |$

$\rightarrow a + (b + c) |)$

Eine Lösung wäre halt das Überspringen der geschlossenen Klammer oder es wird keine geschlossene Klammer automatisch erstellt.

Falls ich es vergessen habe, Dateien und Ordner sollten nach der Erstellung noch verschiebbar sein, eventuell via Drag and Drop.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:**Gab es beim Erstellen von Dateien und Ordnern Probleme?**

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Bitte die erstellten Ordner und Dateien und Projekte alphabetisch sortieren

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese lesen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Nach einer Weile war die gesamte Testgruppe out of sync. Änderungen innerhalb der Datei wurden bei keinem übernommen. Dateien erstellen und löschen und Notifications werden allerdings übernommen. Nachdem noch jemand anderes dem Projekt gejoined ist funktioniert es auf einmal wieder.

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:**Gab es beim Erstellen von Dateien und Ordnern Probleme?**

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

ein klick wäre besser

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese lesen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Neuladen hat nicht geholfen

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

farben am Anfang einmalig festlegen

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese lesen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Am Ende ging gar nix, auch nicht mit neu laden

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Mehr bunt

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Wir haben in unserer Gruppe versucht einen Ordner mit zwei Leuten gleichzeitig zu löschen.

Ein Mitglied meiner Gruppe wurde aus dem Projekt geworfen und konnte danach das Projekt nicht mehr öffnen.

Nach erneutem Laden der Seite ging alles wieder.

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Die geöffnete Datei wird links hervorgehoben, allerdings ist diese dann nicht mehr richtig eingerückt. Das hat Anfangs für ein wenig verwirrung gesorgt.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweise zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Da im folgenden kein Anmerkungsfeld ist, schreibe ich meine Anmerkung hier hinein.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

Ja!

Bei der Verifikation von Car.java wurde bei den gasUp beweisen nicht korrekt dargestellt welchen Index die einzelnen Beweise haben. Es stand immer contract-0 dabei.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Es wäre schön die Cursor der anderen Personen zu sehen

Sonstige Anmerkungen

Positiv:

- Es ist sehr schön die Einzelnen Beweise links neben der Zeilenzahl ausführen zu können.

Verbesserungsvorschläge:

- Wenn man am Dateiende ist wäre es schön trotzdem weiter Herunterscrollen zu können.

Negativ:

- Es wäre schön Dateien und Ordner bewegen zu können um das Projekt leichter umstrukturieren zu können.
-

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Beim Beweisen von mehreren Proof Obligations musste ich mich kurz orientieren

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Dieser Teil des Projekts hat bei mir nicht funktioniert

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Generell wirkt die Reihenfolge der Optionen im Kontextmenü "unkonventionell". Das verlangsamt einen stark.

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Siehe oben

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Optionen für Beweise sind nicht leicht zu finden

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Ich habe es gar nicht bemerkt

Sonstige Anmerkungen

Für eine frühe Version sehr gut. Ich bin optimistisch!

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Einrückung einer neu erstellten Datei (welche wohl automatisch geöffnet wird) hat zunächst für Verwirrung gesorgt, ob diese im richtigen Ordner platziert ist.
--> Vielleicht noch wie bei Editoren üblich die aktive Datei als Tab oder so darstellen.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Vielleicht zusätzlich zum Starten per Button am Rand noch Rechtsklick auf Methodennamen zum Starten ermöglichen (ähnlich wie in KeY?).

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Highlighting ist nicht großartig aufgefallen (lag vermutlich am kurzen Zeitrahmen), stelle ich mir prinzipiell aber sehr hilfreich vor.

Sonstige Anmerkungen

Bei Open Goals: Text zu den einzelnen Zielen kann man nicht komplett lesen, da diese zu lang für Seitenleiste sind.

Vielleicht noch ne Möglichkeit einbauen, um weitere Hinweise zu dem fehlgeschlagenen Beweis anzuzeigen.

Automatische Codevervollständigung wäre noch ein schönes Feature.

Anzeigen des Mauszeigers des Anderen und evtl. was die Person markiert hat, hat gefehlt.

Download funktioniert noch nicht.

Upload funktioniert nicht so recht (funktioniert anscheinend nur bei keinem geöffneten Projekt, selbst da wird zwar Text übernommen, aber keine neue Datei erstellt).

Upload per Drag-and-Drop wäre auch nett.

Finden der Projekte etwas unübersichtlich bei so vielen bereits erstellen Projekten.

Sieht ansonsten top aus! :)

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Habe im editor den punkt zum starten erst seht spät gesehen. Könnte vielleicht ein Punkt mit durchgängiger greller Farbe sein

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Hab es kaum registriert, aber im Augenwinkel war es gut, weil man wusste wo ungefähr der andere gleichzeitig geschrieben hat.

Sonstige Anmerkungen

- leider kein Verschieben von Files und Ordner möglich
 - wenn Fenster mit Ergebnissen mehrerer Beweise angezeigt wurden, wäre ein Button vielleicht ganz gut um alle gleichzeitig zu schließen
 - autoformatting mit gebräuchlicher tastenkombination
 - beim kommentare schreiben mit /** ist es gut, dass bei enter die nächste zeile automatisch ein * hat
 - highlighting von java standard dingen (ohne key annotationen) leider alle in ähnlichem lila
 - leider nicht mehrere Projekte gleichzeitig anzuzeigen möglich
 - beim kollaborativen leider nicht mauszeiger des anderen sichtbar
 - markierungen des jeweiligen anderen nicht sichtbar (wäre cool falls man in einem raum ist kann der eine den code markieren, was dazu sagen und der andere wüsste sofort was er meint)
 - download funktion
 - keine Hinweise was das für ein Programm ist (About page)
 - vielleicht eine funktion, dass man links in der leiste die open goals unter die Projektstruktur ziehen kann (sowas wie bei eclipse, intellij und co). Dann wäre beides gleichzeitig sichtbar
 - vielleicht funktion, dass man ein programm laufen lassen, testen lassen kann (aber liegt vielleicht nicht in eurem Bereich weil keiner wirklich darin programmiert)
 - Tastenkombinationen für wichtigste Funktionen, wie z.B. starten aller beweise, welche dann bei zugehörigen buttons angezeigt werden (siehe andere Programme)
 - syntaxcheck ist sehr gut
-

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:**Gab es beim Erstellen von Dateien und Ordnern Probleme?**

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Die Sortierung von Projekten und Dateien schien mir keinem kleren Muster zu folgen, was zu einer kurzen Suche nach dem passenden Projekt führte.

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Mein erster Schritt war im Menü "Key" nach einer entsprechenden Funktion zu suchen. Dort gab es aber nur alle Beweise ausführen. Mein nächster Schritt war ein Rechtsklick auf die Funktion und den Vertrag, was auch nicht zu einem ergebnis führte. Bei diesem Versuch hab ich allerdings die Breakpoint ähnliche Funktion gefunden den Beweis durchzuführen.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Teilweise wurden Codeabschnitte hervorgehoben, teilweise aber auch nicht.

Sonstige Anmerkungen

Im Editor hat mir das automatische Vervollständigen von Variablen gefehlt.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Das hat wirklich sehr gut geklappt!

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Einrückung für aktive Datei war etwas verwirrend (wirkt wie Unterordner)

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Hab anfangs den kleinen Punkt zum Beweisen der einzelnen Funktionen übersehen.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Editor hat super funktioniert, Anzeigen der Mitarbeitenden Leute ist mir sehr positiv aufgefallen. Evtl. Cursor der anderen Teilnehm

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

Schickes design, evtl. Markierung von Text-Labels im Interface abschalten um 'native-feel' zu erzeugen. Doppelklick zum Öffnen des Projekts zulassen ;).

KeY-Integration hat erstaunlich einfach funktioniert (in Eclipse ist das ja so ne Sache...)!

Weiterhin viel Erfolg!

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:**Gab es beim Erstellen von Dateien und Ordnern Probleme?**

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

Projekte suchen wäre praktisch

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Der kleine blaue Kreis ist leicht zu übersehen

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sich durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Sonstige Anmerkungen

das kollaborierte Arbeiten hat sehr gut funktioniert

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

musste erst rausfinden dass man rechtsklicken kann (etwas unintuitiv für Websites), anschließend aber sehr gut nutzbar

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

habe kurz gebraucht bis ich rausgefunden habe dass man doppelklicken muss, anschließend aber auch gut nutzbar

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

das Blau der kleinen Kreise ist etwas (zu) unauffällig, aber wenn man weiß wo man hinschauen soll ist's klar

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

Highlighting war auch sehr schnell da, perfekt so

Sonstige Anmerkungen

gut und produktiv nutzbar!

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der Nutzbarkeit KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite täglich mit KeY
-
-

Projektverwaltung

Gab es beim Erstellen von Projekten Probleme? *

- Ich hatte keine Probleme beim Erstellen von Projekten.
- Nach kurzer Orientierung konnte ich Projekte problemlos erstellen
- Ich konnte Projekte nur mithilfe eines Betreuers erstellen
- Das Erstellen von Projekten war mir nicht möglich.

Anmerkung:

Anbieten von Create Project bei Rechtsklick auf freie Fläche in Project Tab wäre schön

Gab es beim Erstellen von Dateien und Ordnern Probleme?

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Anmerkung:

Einrücken als Hervorhebung lässt irrtümlich eine tiefer Hierarchieebene vermuten

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme?

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Anmerkung:

Gab es beim Öffnen von Projekten und Dateien Probleme?

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Anmerkung:

KeY

Gab es beim Starten von Beweisen zu einzelnen Methodenverträgen Probleme?

- Ich hatte keine Probleme beim Starten einzelner Beweise
- Nachdem ich die Unterschiede zur bisherigen KeY Oberfläche identifiziert hatte, konnte ich Beweise zu einzelnen Methodenverträgen starten.
- Mir war das Starten von Beweisen zu einzelnen Methoden nur mithilfe eines Betreuers möglich.
- Das Beweisen einzelner Methodenverträge war mir nicht möglich.

Anmerkung:

Outline mit Klassen und Methodennamen wäre schön, so dass man in der Outline die Beweise starten kann.

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme?

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
 - Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
 - Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.
-
-

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien?

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Anmerkung:

Hat das Highlighting von Code anderer Teilnehmer beim kollaborativen Arbeiten unterstützt?

- Das Highlighting hat mir beim kollaborativen Arbeiten geholfen die Übersicht zu behalten
- Das Highlighting hat mir nicht geholfen die Übersicht zu behalten, es war jedoch auch nicht kontraproduktiv
- Das Highlighting hat mich beim kollaborativen Arbeiten gestört.

Anmerkung:

persistente history ("git blame")

Sonstige Anmerkungen

Sehr schöne Applikation! Meine zwei Wünche wären noch ein helleres Theme und eine etwas standardmäßigeren Anordnung der Einträge in den Pull Down Menus

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluation of KollaborierbaRs usability

You're almost done. Now all that is left is to record your experience in this questionnaire. We ask you to answer all questions honestly.

Prior knowledge

*

- I am a student and I am attending the lecture "Formale Methoden im Software Entwurf"/"Formal Methods in Software Engineering" or have done so in the past.
 - I am an employee at TU Darmstadt and I am working with KeY or on improving KeY.
-
-

Editing projects

Did you encounter problems while creating projects? *

- I had no problems creating projects.
- I had to get familiar with the interface briefly, but then I was able to easily create projects.
- I was able to create projects, but only with the help of a supervisor.
- I was not able to create projects.

Remarks:

Did you encounter difficulties while creating files or folders?

- I had no problems creating files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily create files and folders.
- I was able to create files and folders, but only with the help of a supervisor.
- I was not able to create files or folders.

Remarks:

Did you encounter difficulties while deleting projects, files or folders?

- I had no problems deleting files, folders, or projects.
- I had to get familiar with the interface briefly, but then I was able to easily delete projects, files or folders.
- I could delete projects, folders, and files only with the help of a supervisor.
- I was not able to delete files or folders.

Remarks:**Did you encounter difficulties while opening projects, files or folders?**

- I had no problems opening projects, files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily open projects, files or folders.
- I could open projects, folders, or files only with the help of a supervisor.
- I was not able to open files or folders.

Remarks:

KeY

Did you encounter problems when proving method contracts individually?

- I experienced no difficulties proving method contracts individually.
- Only after identifying the differences to the "old" KeY interface, I was able to prove methods individually.
- I could prove individual methods only with the help of a supervisor.
- I was not able to prove methods individually.

Remarks:

Was it difficult to evaluate whether a proof was successful?

- KollaborierbaRs UI made it clear to me, whether a proof was successful or not.
 - I could tell if a proof was successful only after consulting a supervisor.
 - I was not able to tell, whether a proof completed successfully or not.
-
-

Collaborative Editing

Did you notice inconsistent states when editing files collaboratively?

- I did not notice any inconsistencies while editing.
- From time to time I noticed an inconsistency. However I was able to correct it by reloading the file.
- Collaborative editing did not work for me.

Remarks:

Code sections written by other users have been highlighted by the editor. Did this support you while collaboratively editing files?

- Yes, it did help me to get an overview of changes to the file.
- It neither supported me nor did it have a negative impact.
- No, it distracted me from the task.

Remarks:

Additional remarks:

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluation of KollaborierbaRs usability

You're almost done. Now all that is left is to record your experience in this questionnaire. We ask you to answer all questions honestly.

Prior knowledge

*

- I am a student and I am attending the lecture "Formale Methoden im Software Entwurf"/"Formal Methods in Software Engineering" or have done so in the past.
 - I am an employee at TU Darmstadt and I am working with KeY or on improving KeY.
-
-

Editing projects

Did you encounter problems while creating projects? *

- I had no problems creating projects.
- I had to get familiar with the interface briefly, but then I was able to easily create projects.
- I was able to create projects, but only with the help of a supervisor.
- I was not able to create projects.

Remarks:**Did you encounter difficulties while creating files or folders?**

- I had no problems creating files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily create files and folders.
- I was able to create files and folders, but only with the help of a supervisor.
- I was not able to create files or folders.

Remarks:

Ein 'onboarding'-hint, dass man in dem Projektmenü rechtsklicken kann wäre vielleicht hilfreich

Did you encounter difficulties while deleting projects, files or folders?

- I had no problems deleting files, folders, or projects.
- I had to get familiar with the interface briefly, but then I was able to easily delete projects, files or folders.
- I could delete projects, folders, and files only with the help of a supervisor.
- I was not able to delete files or folders.

Remarks:

Did you encounter difficulties while opening projects, files or folders?

- I had no problems opening projects, files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily open projects, files or folders.
- I could open projects, folders, or files only with the help of a supervisor.
- I was not able to open files or folders.

Remarks:

Ordner gingen bei mir of unbeabsichtigt auf und zu

KeY

Did you encounter problems when proving method contracts individually?

- I experienced no difficulties proving method contracts individually.
- Only after identifying the differences to the "old" KeY interface, I was able to prove methods individually.
- I could prove individual methods only with the help of a supervisor.
- I was not able to prove methods individually.

Remarks:

Der Punkt neben der Methode ist an sich cool, aber echt unauffällig

Was it difficult to evaluate whether a proof was successful?

- KollaborierbaRs UI made it clear to me, whether a proof was successful or not.
 - I could tell if a proof was successful only after consulting a supervisor.
 - I was not able to tell, whether a proof completed successfully or not.
-
-

Collaborative Editing

Did you notice inconsistent states when editing files collaboratively?

- I did not notice any inconsistencies while editing.
- From time to time I noticed an inconsistency. However I was able to correct it by reloading the file.
- Collaborative editing did not work for me.

Remarks:

Bei gleichzeitigem Tippen in der gleichen Zeile werden die Edits stark vermischt

Code sections written by other users have been highlighted by the editor. Did this support you while collaboratively editing files?

- Yes, it did help me to get an overview of changes to the file.
- It neither supported me nor did it have a negative impact.
- No, it distracted me from the task.

Remarks:

Additional remarks:

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Name der Seite sollte nicht "React App" sein

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Ist auch nicht aufgegangen

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Kontextmenü sollte sich öffnen beim Klicken in die Zeile der Datei/Ordner etc., nicht nur auf den Namen. Beim Klick auf "src" ist das etwas umständlich, genau das "src" zu treffen.

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Prima

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Rotes Kreuz bei nicht geschlossen an der Editorseite waere noch schoen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Ich hatte zwischendurch das Gefühl, dass sich mein Cursor verschoben hatte. Das kann aber auch an meinem Ungeschick liegen.

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Ich habe ihn nicht benoetigt, da ich es an der offenen Sequenz schon gesehen hatte

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Mein KollaborierbaR Partne rund ich haben es im ersten Anlauf geschafft :-)

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Hatten wir nicht benoetigt. Fuer einen KeY Kenner ist die Konsole nuetzlich.

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Hat Spass gemacht :-)

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Die Vervollständigungsfunktion schlägt noch keine JML keywords vor

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Teilweise kryptische Fehlerbeschreibung

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sehr praktisch!

Sonstige Anmerkungen

Hat sich sehr gut entwickelt! Weiter so!

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Das Problem waren jedoch zu wenige Kenntnisse zu KeY und Beweisbäumen
allgemein bei mir selbst

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

siehe oben

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Nur aktuelle Konsolenausgaben wären besser

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Wenn man einen sehr langen Namen für eine zu erstellende Datei eingibt und dann versucht, die Datei zu erstellen, wird man aus dem Projekt geschmissen ("No open project").

Angenehm wäre auch eine Fehlermeldung, wenn man versucht, eine Datei mit einem Namen zu erstellen, den es schon im selben Verzeichnis gibt. Bisher passiert da einfach nichts, was manche vielleicht verwirren könnte.

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Der eine Syntax-Fehler, der angezeigt wurde, hat geholfen. Dieser blieb dann aber dauerhaft in der Konsole geschrieben und es wurde nichts neues angezeigt. Bei neuen/anderen Fehlern hat sich der Inhalt der Konsole nicht verändert.

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Sie ist nicht Key- und Java-spezifisch genug.

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Löscht die bitte nach jedem Komplizieren

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Die sortierung von Dateien und Ordner im Projekt scheint noch nicht richtig zu funktionieren. Dateien und Ordner werden nicht richtig von einander getrennt, Ordner sind nicht über den Dateien, die Dateien werden nicht richtig sortiert. In meinem Fall waren es folgende Dateinamen: Main.java Main3.java Main2.java , sie wurden in dieser Reihenfolge angezeigt.

Die Dateien und Ordner werden immer noch mit unterschiedlich viele Klicks geöffnet (Dateien werden bei euch mit zwei Klicks, Ordner mit einem Klick geöffnet). Es ist praktischer wenn beide mit der gleichen Anzahl an Klicks geöffnet werden.

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Aktuell beziehen die KeY-Beweise alle Dateien des Projekts oder des Paketes mit ein. Gerade wenn mehrere Leute gleichzeitig an einem Projekt arbeiten ist es praktischer, wenn nur die geöffnete Datei genutzt wird. Sonst gibt es Fehler weil jemand in einer anderen Datei arbeitet und dieser noch nicht compilierfähig ist.

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

- Ja
- Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

- Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.
- Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Die offenen Beweise müssen erst durch aufklappen aller Unterpunkte im Beweisbaum gesucht werden. Der Produktivität halber sollten die offenen Beweise bereits sichtbar / geöffner sein.

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Es sollte nur die letzte Fehlernachricht in der Konsole angezeigt werden. Gerade muss jedes mal durch die gesamte Konsole gescrollt werden. Gerade wenn länger an einer Datei gearbeitet wurde beansprucht das zu viel Zeit.

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Die Einrückungen innerhalb einer Datei sind verbuggt. Wenn in einer eingerückten Zeile Enter gedrückt wird sollte die neue Zeile bereits wie die vorherige Eingerückt sein.

Wenn versucht wird ein Projekt zu erstellen, welches bereits existiert, dann sollte es eine Warnung geben / es sollte das Projekt geöffnet werden. Aktuell passiert absolut garnichts / es gibt keine Warnung / Rückmeldung.

Es gab Probleme mit KeY bei der Nutzung von Paketen.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

JML wird nicht unterstützt

Vervollständigung verändert teilweise ungewollt den Code

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

Das X zum schließen der Goals ist etwas falsch positioniert

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Warte hinweis wenn Proof noch läuft

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Evtl löschen der Fehler in der Konsole, wenn der Beweis fehlerfrei läuft

Sonstige Anmerkungen

Nice App

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

lag aber nicht an dem Programm, sondern an meinen KeY-Fähigkeiten

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

lag aber wieder an den eigenen KeY-Fähigkeiten

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

allerdings nicht immer geleert, wäre hilfreich beim Start des Verifizierens den vorherigen Inhalt zu löschen

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Vervollständigung hat nur für Java funktioniert

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

evtl. Zeilen in 'Open Goals' Tab umbrechen zur einfachereren Lesbarkeit

Sonstige Anmerkungen

Farbliche Markierung des Textes beim kollaborativen Schreiben hat zu etwas unleserlichem Text geführt.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Wär noch cool, wenn man die Konsole selbst leeren könnte

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese ließen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Oben direkt ein Button zum starten aller Beweise statt Menüeintrag.

X zum schließen der sequenten sogeriert nicht , dass man danach zurück zum java file kommt.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluierung der zweiten Nutzerstudie KollaborierbaRs

Fast geschafft. Nun müssen Sie nur noch Ihre Erfahrungen in diesem Fragebogen festhalten. Wir bitten Sie alle Fragen ehrlich zu beantworten.

Vorwissen

Vorwissen *

- Ich bin Informatikstudent und höre derzeit das Modul "Formale Methoden im Software Entwurf"
 - Ich bin Mitarbeiter der TU Darmstadt und arbeite regelmäßig mit KeY
-
-

Aufgabe 1: Wiederholung Projektmanagement

Gab es beim Erstellen von Dateien und Ordnern Probleme? *

- Ich hatte keine Probleme beim Erstellen von Dateien und Ordnern.
- Nach kurzer Orientierung konnte ich Dateien und Ordner problemlos erstellen.
- Ich konnte Dateien und Ordner nur mithilfe eines Betreuers erstellen.
- Das Erstellen von Dateien oder Projekten war mir nicht möglich.

Gab es beim Löschen von Projekten, Ordnern und Dateien Probleme? *

- Ich hatte keine Probleme beim Löschen von Dateien, Ordnern und Projekten.
- Nach kurzer Orientierung konnte ich Projekte, Ordner und Dateien löschen.
- Ich konnte Projekte, Ordner und Dateien nur mithilfe eines Betreuers löschen.
- Das Löschen von Projekten, Ordnern und Dateien war mir nicht möglich.

Gab es beim Öffnen von Projekten und Dateien Probleme? *

- Ich hatte keine Probleme beim Öffnen von Projekten und Dateien.
- Nach kurzer Orientierung konnte ich Projekte und Dateien öffnen.
- Ich konnte Projekte und Dateien nur mithilfe eines Betreuers öffnen.
- Das Öffnen von Projekten und Dateien war mir nicht möglich.

Hat sich die neue Sortierung der Projektstruktur als hilfreich erwiesen?

Ja

Nein

Hat sich das umsortierte Kontextmenü als hilfreich erwiesen?

Ja

Nein

Anmerkung:

KeY

Gab es beim Interpretieren der Rückmeldung über den Erfolg eines Beweises Probleme? *

- Ich konnte anhand der Rückmeldung von KollaborierbaR immer sofort sagen, ob ein Beweis erfolgreich war oder nicht.
- Erst nach Erläuterung eines Betreuers konnte ich nachvollziehen, ob ein Beweis fehlgeschlagen ist oder nicht.
- Ich konnte nicht nachvollziehen, ob ein bestimmter Beweis fehlgeschlagen ist oder nicht.

Anmerkungen

Kollaboratives Editieren

Kam es zu inkonsistenten Zuständen beim kollaborativen Bearbeiten von Dateien? *

- Es kam zu keinen inkonsistenten Zuständen beim kollaborativem Editieren.
- Es kam gelegentlich zu inkonsistenten Zuständen. Diese liessen sie durch Neuladen der Datei beheben.
- Kollaboratives Editieren in KollaborierbaR war nicht möglich. Die Anwendung ist bei jeglichen Inkonsistenzen abgestürzt.

Die neue Vervollständigungsfunktion hat sich als hilfreich erwiesen?

Ja

Nein

Anmerkungen

Aufgabe 2: Fehlerhafte Spezifikation

Anmerkung:

Hat Ihnen der Beweisbaum beim Finden des Fehlers geholfen? *

Ja, der Beweisbaum hat beim Finden des Fehlers in der Spezifikation geholfen.

Nach einer kurzen Erklärung des Beweisbaums durch einen Betreuer hat mir der Beweisbaum beim Finden des Fehlers geholfen.

Nein, der Beweisbaum hat mir nicht geholfen, den Fehler in der Spezifikation zu finden.

Anmerkung:

Lag vermutlich an dem Problem an sich, nicht daran, dass es zu unübersichtlich ist.

Hat Ihnen das Anzeigen des Sequenten beim Finden des Fehlers geholfen? *

- Ja, das Anzeigen des Sequenten hat beim Finden des Fehlers in der Spezifikation geholfen.
- Nach einer kurzen Erklärung der Sequentenanzeige hat mir diese geholfen, den Fehler in der Spezifikation zu finden.
- Nein, das Anzeigen des Sequenten hat mir nicht beim Finden des Fehlers in der Spezifikation geholfen.

Anmerkung:

Aufgabe 3: Suche im Array

Hat Ihnen der Beweisbaum und das Anzeigen des Sequenten beim Verifizieren Ihres Algorithmus geholfen?

- Ja, mithilfe des Beweisbaums und dem Sequent konnte ich meine Spezifikation stückweise verbessern und letztlich beweisen.
- Mithilfe des Betreuers konnte ich durch den Beweisbaum und den Sequenten meine Spezifikation verifizieren.
- Der Beweisbaum und der Sequent konnte mir nicht beim Spezifizieren und Verifizieren meines Algorithmus helfen.

Anmerkungen:

Hat Ihnen bei der Bearbeitung der Spezifikation die Anzeige der KeY Fehler innerhalb der Konsole geholfen? *

- Ja, die KeY Fehler innerhalb der Konsole haben mir beim Schreiben der Spezifikation geholfen.
- Nein, die KeY Fehler innerhalb der Konsole haben mir nicht beim Schreiben der Spezifikation geholfen.
- Mir ist die Konsole nicht aufgefallen

Anmerkung:

Haben Sie generelle Anmerkungen zum Arbeitsfluss mit KollaborierbaR bezüglich des Beweisbaums und des Anzeigens des Sequenten?

Sonstige Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Evaluation of the second user study KollaborierbaRs

Almost there. Now all you have to do is record your experiences in this questionnaire. We ask you to answer all questions honestly.

Prior knowledge

Prior knowledge *

- I am a student and I am attending the lecture "Formale Methoden im Software Entwurf"/"Formal Methods in Software Engineering" or have done so in the past.
 - I am an employee at TU Darmstadt and I am working with KeY or on improving KeY.
-
-

Task 1: Editing Projects

Did you encounter difficulties while creating files or folders? *

- I had no problems creating files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily create files and folders.
- I was able to create files and folders, but only with the help of a supervisor.
- I was not able to create files or folders.

Did you encounter difficulties while deleting projects, files or folders? *

- I had no problems deleting files, folders, or projects.
- I had to get familiar with the interface briefly, but then I was able to easily delete projects, files or folders.
- I could delete projects, folders, and files only with the help of a supervisor.
- I was not able to delete files or folders.

Did you encounter difficulties while opening projects, files or folders? *

- I had no problems opening projects, files or folders.
- I had to get familiar with the interface briefly, but then I was able to easily open projects, files or folders.
- I could open projects, folders, or files only with the help of a supervisor.
- I was not able to open files or folders.

Did the new sorting of the project structure prove to be helpful? *

Yes

No

Did the reordered context menu prove helpful?

Yes

No

Remarks

KeY

Was it difficult to evaluate whether a proof was successful? *

KollaborierbaRs UI made it clear to me, whether a proof was successful or not.

I could tell if a proof was successful only after consulting a supervisor.

I was not able to tell, whether a proof completed successfully or not.

Remarks

Collaborative Editing

Did you notice inconsistent states when editing files collaboratively?

*

- I did not notice any inconsistencies while editing.
- From time to time I noticed an inconsistency. However I was able to correct it by reloading the file.
- Collaborative editing did not work for me.

Has the new completion function proven to be helpful?

- Yes
- No

Remarks

Task 2: Incorrect specification

Did the proof tree help you to find the error? *

- Yes, the proof tree helped me to find the error in the specification.
- After a short explanation of the proof tree by a supervisor, it helped me to find the error.
- No, the proof tree didn't help me to find the bug in the specification.

Remarks

Did the display of the sequent help you to find the error?

- Yes, displaying the sequent helped to find the error in the specification.
 - After a short explanation of the displayed sequent it helped me to find the error in the specification.
 - No, displaying the sequent did not help me to find the error in the specification.
-
-

Task 3: Searching an array

Did the proof tree and the display of the sequent help you to verify your algorithm? *

- Yes, with the help of the proof tree and the sequent I was able to improve my specification bit by bit and finally prove it.
- With the help of the supervisor, I was able to verify my specification through the proof tree and the sequents.
- The proof tree and the sequence could not help me to specify and verify my algorithm.

Remarks

Did the display of the KeY errors within the console help you while editing the specification? *

- Yes, the KeY errors displayed in the console helped me to write the specification.
- No, the KeY errors displayed in the console didn't help me write the specification.
- I didn't notice the console.

Remarks:

Do you have any general comments on the workflow with KollaborierbaR regarding the proof tree and the display of the sequents?

Additional remarks

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

Code Reviews: Ausgefüllte Checklisten

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

Sidebar

Betroffene User Stories *

Anzeigen von Ordnern/Dateien innerhalb der Sidebar

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Anlegen von Dateien

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Die Klasse ProjectController.java auf dem Server ist sowohl für die Netzwerkschnittstelle als auch für den Zugriff auf das Dateisystem zuständig.

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Bitte ProjectController.java in Netzwerk und Dateisystem aufspalten

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Anlegen von Dateien

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

ProjectController.java ist nicht mehr für den Zugriff auf das Dateisystem zuständig. Die Funktionalität wurde in Memory.java ausgelagert. Damit kann der Merge Request angenommen werden.

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Erzeugen eines Projekts

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

FileUpAndDownload

Betroffene User Stories *

File download

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

An der Spezifikation wurde nichts geändert.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

FileUpAndDownloaD

Betroffene User Stories *

File upload

Wieviele Durchführung des Review? *

- 1 2 3 4
-

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Spezifikation wurde nicht verändert.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

jml-highlighting

Betroffene User Stories *

Jml Kommentar und Keyword Highlighting, Java Highlighting ohne Rottoene

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Löschen von Dateien

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

React Lifecycle Methoden nicht zu dokumentieren ist meiner Ansicht nach in Ordnung.

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Lösichern von Ordnern

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

not-supported-errors

Betroffene User Stories *

Not supported errors

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

openfile

Betroffene User Stories *

Oeffnen von Dateien in der Weboberflaeche

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

openprojectmodal

Betroffene User Stories *

Projekt Menu Funktion - Open Project

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.500 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Anlegen von Ordnern

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Löschen von Projekten

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

sidebar-synchronisation

Betroffene User Stories *

Automatische Synchronisation der Projektdarstellung in der Sidebar

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.515 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

key-proof

Betroffene User Stories *

Einfaches Beweisen von ProofObligations mit Button in der UI

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

Projectmanagement

Betroffene User Stories *

Speichern von Dateien innerhalb der Webanwendung

Wieviele Durchführung des Review? *

- 1 2 3 4
-

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

projectmanagement

Betroffene User Stories *

Umbenennen von Dateien, Ordnern und Projekten

Wievielte Durchführung des Review? *

1	2	3	4
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

key-proof

Betroffene User Stories *

Beweis Starten

Wievielte Durchführung des Review? *

1	2	3	4
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Das Akzeptanzkriterium umfasst ein weiteres schwer umzusetzendes Feature das nicht implementiert wurde. Dies wurde jedoch mit dem Auftraggeber abgesprochen und als nicht notwendig klassifiziert.

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach
ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der
Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.),
wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen
Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne
Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

Text collaboration

Betroffene User Stories *

Markierung Kollaborativer Editierungen, Kollaboratives_Editieren

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Websocket Spezifikation fehlt

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Server Synchronization teils duerftiges javadoc

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Kommentare und Spezifikation müssen nachgebessert werden

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

Text collaboration

Betroffene User Stories *

Markierung Kollaborativer Editierungen, Kollaboratives_Editieren

Wievielte Durchführung des Review? *

1	2	3	4
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
--------------------------	--------------------------	--------------------------	--------------------------	-------------------------------------

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

user-names

Betroffene User Stories *

Nutzernamen vergeben

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

goallist-ui

Betroffene User Stories *

Offene Goals anzeigen

Wievielte Durchführung des Review? *

1	2	3	4
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

select-obligations

Betroffene User Stories *

Statusanzeigen Proof Obligations

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Spezifikation wurde nicht verändert.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.565 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

createbeforedelete

Betroffene User Stories *

AnpassungMenüsortierung

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

file-icon-test schlägt fehl

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

createbeforedelete

Betroffene User Stories *

AnpassungMenüsortierung

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

autocompletion

Betroffene User Stories *

Autovervollstaendigung

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

wurde nicht erweitert

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

improved-modal

Betroffene User Stories *

Doppelklick auf Projektnamen öffnet Projekte

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Diese Userstory wurde als Reaktion auf die erste Nutzerstudie angefertigt und bearbeitet.

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.585 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

indentation

Betroffene User Stories *

Einheitliche einrueckung Dateistruktur

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Es wurde nichts an der Spezifikation verändert.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

indentation

Betroffene User Stories *

Scrollen innerhalb Opengoal anzeigen

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Es wurde nichts an der Spezifikation geändert.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

sortierteprojekte

Betroffene User Stories *

Sortierteprojekte

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

ctrl-z

Betroffene User Stories *

Steuerung+Z

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Benutzt keine API

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

proofTree

Betroffene User Stories *

Anzeigen der Beweisbaumstruktur unter den Open Goals

Wievielte Durchführung des Review? *

1	2	3	4
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Es liegen keine Veränderungen der Spezifikation vor.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

sequents

Betroffene User Stories *

Anzeigen eines Sequenten

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.616 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

console

Betroffene User Stories *

console

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

A.621 c (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Props validation fehlt in console.jsx

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Missing comments

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

console

Betroffene User Stories *

console

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

foldersbeforefiles

Betroffene User Stories *

Sortierung Sidebar

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Martin Kerscher ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Es wurden keine neuen Schnittstellen definiert oder Schnittstellen verändert

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Die Parameter und Rückgabewerte der Funktion `sort` sind nicht gesondert dokumentiert aber werden aus der Dokumentation ersichtlich

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

methodRightClick

Betroffene User Stories *

Beweisen von Methoden mit Rechtsklick

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

PeiltastenMergeProbleme

Betroffene User Stories *

Pfeiltasten

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

Bei manchen Methoden fehlt '@' im TSDoc

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

PfeilsatenMergeProbleme

Betroffene User Stories *

Pfeiltasten

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Jonas Belouadi ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

collabProvingDocumentation

Betroffene User Stories *

SB1103 - Synchrone Beweise

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

database

Betroffene User Stories *

Speichern von Beweisen

Wievielte Durchführung des Review? *

1	2	3	4
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

make check schlaegt fehl

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

an der Spezifikation gab es keine Änderungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

File.java nicht kommentiert

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Nicht bestanden, make pipeline laeuft nicht durch, teils fehlende Kommentare,

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

database

Betroffene User Stories *

Speichern von Beweisen

Wievielte Durchführung des Review? *

1	2	3	4
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

Pipeline läuft nun erfolgreich durch.

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Test laufen aufgrund von neueren Tests nicht mehr durch.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Es sind nun alle Dateien dokumentiert.

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Nein, wurde nicht bestanden die api tests laufen nicht durch.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

database

Betroffene User Stories *

Speichern von Beweisen

Wievielte Durchführung des Review? *

- 1 2 3 4
-

Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold	Jonas Belouadi	Anton Haubner	David Heck	Martin Kerscher
-------------	----------------	---------------	------------	-----------------

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind , dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessens des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Es laufen nun auch die neueren Tests durch.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung)).

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht.

Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Keine Maengel mehr gefunden.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt.

Google Formulare

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

proofHistoryStub

Betroffene User Stories *

SvB1103 - Speichern von Beweisen

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

Marc Arnold



Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

- JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)
TSDoc (<https://github.com/Microsoft/tsdoc>)
JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

macros

Betroffene User Stories *

Ausfuehren von Proofscript, Highlighting Proofscript

Wievielte Durchführung des Review? *

1 2 3 4

Reviewer *

David Heck ▾

Verantwortliche(r) für Features *

Marc Arnold Jonas Belouadi Anton Haubner David Heck Martin Kerscher

Entwickler

Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

make check schlaegt fehl

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Spezification muss noch gemacht werden

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

ProofCollabController.ts nicht kommentiert

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Leider gab es bei Key ein update, welches dazu gefuehrt hat das ein Teil der Funktionalitaet nicht mehr funktioniert.

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

marcos

Betroffene User Stories *

Ausfuehren von Proofscript, Highlighting Proofscript

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

Make pipeline läuft nun inklusive make check durch.

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

Spezifikation nun vorhanden und getestet.

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Unkommentierte Datei war nicht Teil dieses Codereviews.

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helpermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

Fehlende Funktionalitaet hatte nichts mit einem Key update zu tun. Der branch im git submodul war lediglich nicht auf dem neusten stand.

This content is neither created nor endorsed by Google.

Google Forms

KollaborierbaR Code Review

Checkliste

Es sind nur Änderungen vom Review betroffen, d. h. alter Code muss nicht auf die genannten Anforderungen geprüft werden.

Betroffener Branch *

welcomeScreen

Betroffene User Stories *

UI Verbesserung ausserhalb von User Story: Welcome Screen

Wievielte Durchführung des Review? *

1

2

3

4



Reviewer *

David Heck



Verantwortliche(r) für Features *

Marc Arnold

Jonas Belouadi

Anton Haubner

David Heck

Martin Kerscher

Entwickler



Anmerkungen

Manueller Funktionstest

- Die Implementierung der Funktionalität erfüllt die Akzeptanzkriterien der zugehörigen User Stories?

Anmerkungen

Überprüfung mittels Buildpipeline

Kann mittels `make pipeline` vollständig ausgeführt werden

- Statische Analyse und Stilprüfung erfolgreich? (`make check`)
- Code kompiliert? (`make`)
- API-Tests erfolgreich? (`make test` auf Server)

Anmerkungen

API Dokumentation

Falls eine Frage nicht zutrifft, da z. B. die API nicht verändert wurde und daher keine Ergänzungen der Spezifikation notwendig sind, dennoch abhaken.

Ergänzung der Spezifikation

- OpenAPI 2.0 Spezifikation ergänzt für alle neuen Serverrouten?
- Ergänzungen enthalten ein Beispiel für Eingabedaten (falls z.B. POST)?
- Ergänzungen enthalten ein Beispiel für Rückgabewert?
- Ergänzungen enthalten Schema der beteiligten Datentypen?
- Ergänzungen enthalten Beschreibungen, die aus Sicht des Reviewers zum Verständnis ausreichen?

Tests

- Es sind (Karate-)Tests vorhanden, die die beschriebene Spezifikation nach ermessen des Reviewers ausreichend testen?
- Der Client-Code nutzt die neuen/veränderten Schnittstellen gemäß der Spezifikation?

Anmerkungen

api nicht veraendert

Dokumentation

Hier gelten folgende Einschränkungen, um den Dokumentationsaufwand in einem realistischen Maß zu halten:

- * Getter, Setter und triviale Konstruktoren müssen nicht dokumentiert werden
- * Parameter und Rückgabewerte müssen nur gesondert dokumentiert werden (@param etc.), wenn sich ihr Zweck aus Sicht des Reviewer nicht bereits aus deren Benennung oder der übrigen Dokumentation ergibt

Dokumentationsformat:

JavaDoc (<https://www.oracle.com/technetwork/articles/java/index-137868.html>)

TSDoc (<https://github.com/Microsoft/tsdoc>)

JSDoc (<http://usejsdoc.org/>)

- Methoden und Klassen sind dokumentiert mittels JavaDoc/TSDoc/JSDoc?
- Sind komplexe Algorithmen kommentiert, die der Reviewer ohne Kommentierung nicht verstehen würde?

Anmerkungen

Objektorientiertes Design

Erfüllen Klassen grob das Single Responsibility Prinzip aus Sicht des Reviewers?

(Eingrenzung von "grob": Eine Klasse sollte nicht gleichzeitig für das KeY-assistierte Beweisen eines Methodenkontrakts und für die Netzwerkschnittstelle zum Abruf des erzeugten Beweises gleichzeitig zuständig sein (grobe Verletzung).)

Das Führen des Beweises und eine Umwandlung in eine Datenstruktur, welche sich zur Übertragung über das Netzwerk eignet, ist hingegen in Ordnung (auch wenn man auch dies als getrennte Verantwortlichkeiten sehen kann.)

- SRP grob erfüllt?

Datenkapselung / Information Hiding wird vorgenommen? (für TypeScript und Java Code)

Z. B. Klassenattribute sind private, sofern es nicht einen guten Grund gibt, der dagegen spricht. Natürlich sind reine Datenklassen die beispielsweise zur Übertragung über das Netzwerk dienen, ausgenommen.

Auch Helfermethoden, welche allein internen Zwecken einer Klasse dienen, sollten private/protected sein.

- Datenkapselung / Information Hiding wurde praktiziert? (TypeScript und Java)

Anmerkungen

Abschluss

In der Regel sollte das Review nur als bestanden gelten, wenn ***alle*** Anforderungen erfüllt wurden. Bei Außnahmen kann dies in den Textfeldern begründet werden.

- Wurde das Code Review bestanden?

Anmerkungen

This content is neither created nor endorsed by Google.

Google Forms