Prof. Dr. Stephan Kleuker
Hochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Objektorientierte Analyse und Design

Sommersemester 2024
11. Aufgabenblatt

Bitte nehmen Sie an der anonymen Lehrevaluation unter https://forms.gle/erJ3qKAscLnbcKkU6 teil. Die von mir kommentierten Ergebnisse stehen im letzten Fragen&Antworten-Dokument des Semesters.

Aufgabe 0.11 (0 Punkte)
Geben Sie das Lösungswort des Quiz aus der Lernnotiz an.

Aufgabe 32 (9 Punkte)

Gegeben sei die von die Veranstaltungsseite erhältliche, gegenüber der Vorlesung leicht erweiterte Redux-Konzeptimplementierung. Führen Sie die folgenden Erweiterungen durch, notieren Sie dabei jeweils, welche Klassen sie wie verändert haben (z. B. „Methode reduce() in Reducer erweitert“) und ob es alternative sinnvolle Lösungen gibt. Geben Sie für das finale Ergebnis das zugehörige Klassendiagramm an.

a) Ergänzen Sie die Möglichkeit über die Nutzungsoberfläche Testdaten einzuspielen. Diese Testdaten liegen in der Klassenmethode TaskList.testdaten() bereits vor. Die Testdaten ersetzen alle bisher vorhandenen Einträge.
b) Ergänzen Sie die Möglichkeit über die Nutzungsoberfläche eine Task als abgeschlossen zu markieren.
c) Ergänzen Sie die Möglichkeit über die Nutzungsoberfläche sich die Tasks für eine verantwortliche Person anzeigen zu lassen.
d) Ergänzen Sie die Möglichkeit über die Nutzungsoberfläche mit einem Befehl eine oder mehrere Task-Ids zu übergeben, so dass danach diese Tasks nicht löschbar sind.
e) Durch die vorhandene Klassenmethode util.User.allowed(String) kann überprüft werden, ob ein Name für eine bearbeitende Person erlaubt ist. Bauen Sie diese Prüfung in einem neuen Store ein, so dass hier bereits unerlaubte Bearbeitungen (durch AddAction) abgefangen werden (welche Lösungsalternative sehen sie?).
f) Ergänzen Sie die Möglichkeit über die Nutzungsoberfläche alle Tasks einer bearbeitenden Person einer anderen erlaubten bearbeitenden Person zuzuweisen.

Das Projekt enthält einen Nutzungsdialog für die beschriebene Funktionalität, ein Beispielablauf sieht wie folgt aus, Eingaben sind umrandet, Ausgaben müssen nicht identisch aussehen.

```
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) Task eines Bearbeiters
(6) Tasks schuetzen
(7) Tasks neuzuordnen
3
Task{id=1, text=Ziele herausfinden, responsible=Ute, finished=false}
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=false}

Dauer von TestAction{}: 1267212
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
```

```
(6) Tasks schuetzen
(7) Tasks neuzuordnen
4
welche Id: 4
Task{id=1, text=Ziele herausfinden, responsible=Ute, finished=false}
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=true}

Dauer von FinishAction{finishId=4}: 137045
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
5
fuer welche Person: Leila
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=true}
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
6
kommaseparierte Liste nicht loeschbarer Tasks: 2, 3
Task{id=1, text=Ziele herausfinden, responsible=Ute, finished=false}
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=true}

Dauer von ProtectAction{ids=[[2, 3]]}: 206171
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
2
welche Id: 2
Task{id=1, text=Ziele herausfinden, responsible=Ute, finished=false}
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=true}
```

```
Dauer von DeleteAction{deleteId=2}: 111387
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
2
welche Id: 1
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Leila, finished=true}

Dauer von DeleteAction{deleteId=1}: 170249
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
7
aktuelle bearbeitende Person: Urs
neue bearbeitende Person: GibsNich
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
7
aktuelle bearbeitende Person: Leila
neue bearbeitende Person: Urs
Task{id=2, text=Stakeholder finden, responsible=Urs, finished=false}
Task{id=3, text=UseCases finden, responsible=Urs, finished=false}
Task{id=4, text=Aktivitaetsdiagramme entwerfen, responsible=Urs, finished=true}

Dauer von RearrangeAction{parameter=[Leila, Urs]}: 201341
(0) beenden
(1) Task hinzu
(2) Task loeschen
(3) Testdaten einspielen
(4) Task abschliessen
(5) zugeordnete Tasks einer Person
(6) Tasks schuetzen
(7) Tasks neuzuordnen
0
```