Die Hausaufgaben müssen von jedem Studierenden einzeln bearbeitet und abgegeben werden. Für die Hausaufgabe sind die aktuellen Informationen vom Blog `https://seblog.cs.uni-kassel.de/ws2021/programming-and-modelling/` zu berücksichtigen.

Abgabefrist ist der 18.02.2021 - 23:59 Uhr

Abgabe

Wir benutzen für die Abgabe der Hausaufgaben Git. Jedes Repository ist nur für den Studierenden selbst sowie für die Betreuer und Korrektoren sichtbar.

Für die Hausaufgabe benötigt ihr das **bestehende MiniChat-Repository**, welches über folgenden Link angelegt werden kann, falls nicht bereits geschehen:

`https://classroom.github.com/a/KokiWtaz`

Nicht oder zu spät gepushte (Teil-)Abgaben werden mit 0 Punkten bewertet.

Abgaben, die nicht kompilieren oder Laufzeitfehler beinhalten, werden mit 0 Punkten bewertet!

Vorbereitungen

Fügt folgende Dateien aus dem .zip-Archiv in euer Projekt ein:

- de.uniks.pmws2021.chat.network.client.Response
- de.uniks.pmws2021.chat.network.client.RestClient
- de.uniks.pmws2021.chat.network.client.WSCallback
- de.uniks.pmws2021.chat.network.client.WebSocketClient
- de.uniks.pmws2021.chat.network.client.CustomWebSocketConfigurator

Aufgabe 0 - Evaluation

Zum Ende der Veranstaltung würden wir gerne eine Evaluation durchführen.

Der folgende Link führt zur Umfrage. Wir bedanken uns im Voraus bei allen, die sich die Zeit nehmen, um uns bei der Gestaltung der Lehre zu helfen.

`https://forms.gle/WGP3dR8t95RiVJAG9`

Aufgabe 1 - RestClient (5P)

In dieser Aufgabe soll die RestClient-Klasse vervollständigt werden. Hierzu dienen die Kommentare in der oben erwähnten Vorlage.

Folgende Methoden müssen implementiert werden:

- `login(String name, Callback<JsonNode> callback)`
  Sendet eine Login-Anfrage an den Server
- `logout(String name, Callback<JsonNode> callback)`
  Sendet eine Logout-Anfrage an den Server
- `getAllAvailableUsers(Callback<JsonNode> callback)`
  Sendet eine Anfrage zu den aktuell angemeldeten Benutzern an den Server

Aufgabe 2 - WebSocketClient (21P)

In dieser Aufgabe soll die WebSocketClient-Klasse vervollständigt werden. Hierzu dienen die Kommentare in der oben erwähnten Vorlage.

Der Konstruktor des WebSocketClient muss folgendes leisten:

- ClientEndpoint konfigurieren

Darüber hinaus müssen folgende Methoden implementiert werden:

- `onOpen(Session session, EndpointConfig config)`
  Wird bei der Registrierung einer neuen Verbindung aufgerufen
- `onClose(Session session, CloseReason closeReason)`
  Wird beim Schließen einer Verbindung aufgerufen
- `onMessage(String message)`
  Verarbeitet eine hereinkommende Nachricht
- `sendMessage(String message)`
  Sendet eine Nachricht an den Server
- `stop()`
  Beendet die Websocket-Session

Aufgabe 3 - UI-Anbindung (34P)

Abschließend soll die Client-Anwendung komplettiert werden. Hierzu muss euer Controller der ClientView (vgl. HA09) angepasst werden.

init()

- Verbindung zum Server aufbauen (Rest Login)
- Mit dem Websocket verbinden
- Alle User, welche momentan auf dem Server online sind, in der Liste anzeigen.
- Eine Methode für den Websocket übergeben, welche die eingehenden Nachrichten wie folgt verarbeitet:
  - Unterscheiden zwischen all- und private-Chatnachrichten, sowie den Systemnachrichten
  - All-Chatnachricht wird im dafür vorgesehenen Tab/Chatfenster angezeigt
  - Private-Chatnachricht wird im Tab/Chatfenster des jeweiligen Benutzers angezeigt
  - Systemnachricht fügt einen Benutzer hinzu oder entfernt diesen. Bei einer joined-Nachricht muss ein Benutzer erstellt, bei einer left-Nachricht muss ein Benutzer entfernt werden.

stop()

- Websocket-Verbindung trennen

Funktionalität

Implementiert zuletzt die Funktionalität aller Elemente des ClientScreens entsprechend der Beschreibungen aus HA 09.

Hinweis:

Das Erzeugen/Entfernen neuer Tabs in der TabPane muss im Code geschehen.

Abschluss

Nach Implementierung der Funktionalität dieser Hausaufgabe ist die Anwendung MiniChat für dieses Semester abgeschlossen. Es muss somit möglich sein, einen Server und mehrere Clients zeitgleich zu starten und unter den Clients Nachrichten zu verschicken.

Anhang

Es folgt eine Auflistung hilfreicher Webseiten und weiterer Erklärungen zu den Themen dieser Hausaufgabe. Die Links sind als Startpunkt zur selbstständigen Recherche angedacht. Das Durcharbeiten der folgenden Quellen ist kein bewerteter Anteil der Hausaufgaben.

Unirest

- Wie benutzte ich Unirest für REST Anfragen:
  `http://kong.github.io/unirest-java/#requests`