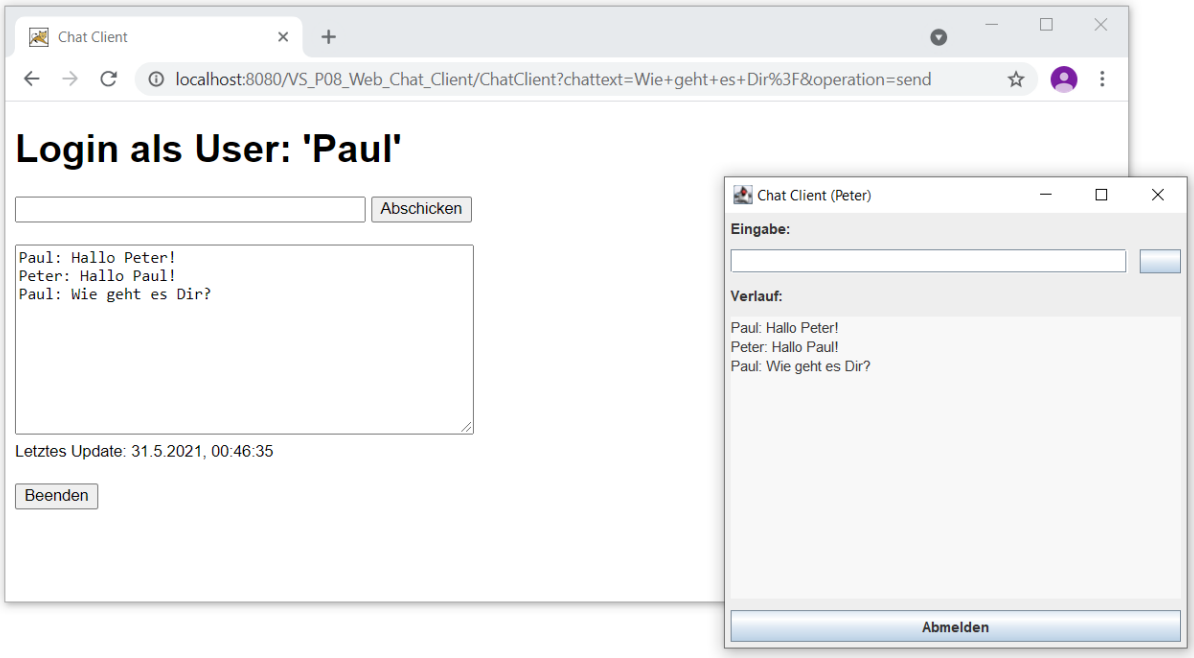
Aufgabenblatt 7

Aufgabe – Web-Client für das Chat-System

Implementieren Sie einen Web-Client für das RMI-basierte Chat-System des vorhergehenden Aufgabenblattes.

Folgende Anforderungen sind zu erfüllen:

- Ein Teil der Anwendung (nämlich der RMI-Client) läuft somit in der Web-Anwendung. Der Chat-Server und die **`rmiregistry`** laufen außerhalb des Web-Containers.
- Der native RMI-Client aus dem vorhergehenden Aufgabenblatt soll weiterhin parallel genutzt werden können.
- Die Realisierung soll als ein oder mehrere Servlets (ggf. noch **`jsp`**- oder **`html`**-Seiten) erfolgen.
- Über ein Formular können Chat-Texte verschickt und die Nachrichten des Chats anzeigt werden. Über dasselbe Formular kann der Benutzer auch aus dem Chat ausgetragen werden.



- Bei der Initiierung des Chats gibt der Benutzer seinen Namen über ein Formular ein, mit dem er beim Chat registriert wird. Unter diesem Namen wird eine **`HttpSession`** erzeugt, die beim Verlassen des Chats invalidiert wird. Parallel zum Erzeugen und Beenden der **`HttpSession`** wird der Client in der RMI-Registry ein- bzw. ausgetragen.
- Im Fehlerfall (Server / Registry kann nicht erreicht werden, Kommunikationsfehler, Session abgelaufen, …) sollen entsprechende Seiten Aufschluss über den aufgetretenen Fehler geben.

Hinweise:

Im Prinzip sind Sie frei, eigene Wege zur Lösung der Aufgabenstellung einzuschlagen. Damit die Aufgabe auch durch Studierende mit geringeren Vorkenntnissen im Bereich der Web-Programmierung behandelt werden kann, werden hier einige Hinweise gegeben.

- Da Sie an verschiedenen Stellen in der Web-Applikation auf die RMI-Komponenten zugreifen müssen, bietet es sich an, an einer Stelle die Verweise auf die entsprechenden Objekte der RMI-Kommunikation zu verwalten. Da die Funktionalität der Anwendung überschaubar ist, kann diese durch **ein einziges Servlet** realisiert werden.

- Unter dieser Prämisse, muss das Servlet auch **parametrisierbar** hinsichtlich der durchzuführenden Operationen (initialisieren, de-initialisieren, Text senden, Nachrichten anzeigen,…) sein. Diese Parametrisierung kann über die Parameter-Listen entsprechender **`GET`**-Requests vorgenommen werden. Hier ist die Nutzung sog. **`hidden`-Felder** in den Formularen sinnvoll. Bsp.:
  `<input type="hidden" name="operation" value="init">`

- Am komplexesten ist die Aktualisierung der Anzeige der empfangenen Chat-Nachrichten. Sie können hierzu die bei der Initiierung erzeugte **`HttpSession`** verwenden. Z.B. kann man in der **Session eine Liste der letzten *n* Nachrichten** speichern. Wird Client-seitig über **`receiveMessage()`** eine Nachricht über das Callback-Objekt empfangen, so wird diese in die Liste im Session–Objekt eingetragen. Beim Anzeigen des Formulars wird die Liste ausgelesen und in ein entsprechendes **`html`**-Element ausgegeben. Für die Ausgabe kann z.B. verwendet werden:
  `<textarea name="chatoutput" cols="50" rows="10"readonly>`

- Den jeweiligen Sessions zuzuordnen sind dabei auch die Verweise auf die Chat-Objekte.

- Achten Sie beim Test auf die unterschiedlichen Möglichkeiten der Realisierung der Session. Wollen Sie von einem Browser aus mit mehreren Client-Sitzungen die Applikation testen, dann müssen Sie **URL-Rewriting** unterstützen. Bei der Verwendung von **Cookies** werden die UI-Instanzen nicht in Form von unterschiedlichen Sessions unterschieden.

- Die Anzeige der empfangenen Nachrichten muss ständig aktualisiert werden. Es gibt zahlreiche Wege dies zu tun. Die einfachste Möglichkeit ist, in **zyklischen Intervallen** (*Polling*) durch Absetzen eines entsprechenden **`GET`**-Requests, die Seite einfach neu zu laden :
  `<meta http-equiv="refresh" content="10;`
  `URL=ChatClient?name=heijo&operation=show">`

- Um dem Client-seitigen RMI-Callback-Objekt, das sich jetzt in der Web-Applikation befindet, Nachrichten zustellen zu können, muss dieses über ein Socket vom Chat-Server angesprochen werden können. Dazu müssen entsprechende **Security Policies** eingestellt werden. Hierzu bietet der Tomcat ein umfangreiches Framework an. Zur Vereinfachung können Sie hier auch die Standard-Policy der Java VM editieren und die in der Vorlesung vorgestellte Test-Policy verwenden:
  `grant { permission java.security.AllPermission; };`

- Eine umfangreiche Sammlung von einsetzbaren **`html`**-Elementen finden Sie unter:
  `http://de.selfhtml.org/`

Für das Testat ist ein Protokoll bekannter Formatierung (siehe Blatt 1) inkl. der durchgeführten Tests vorzulegen.

Testierung: 7./8.6.2021