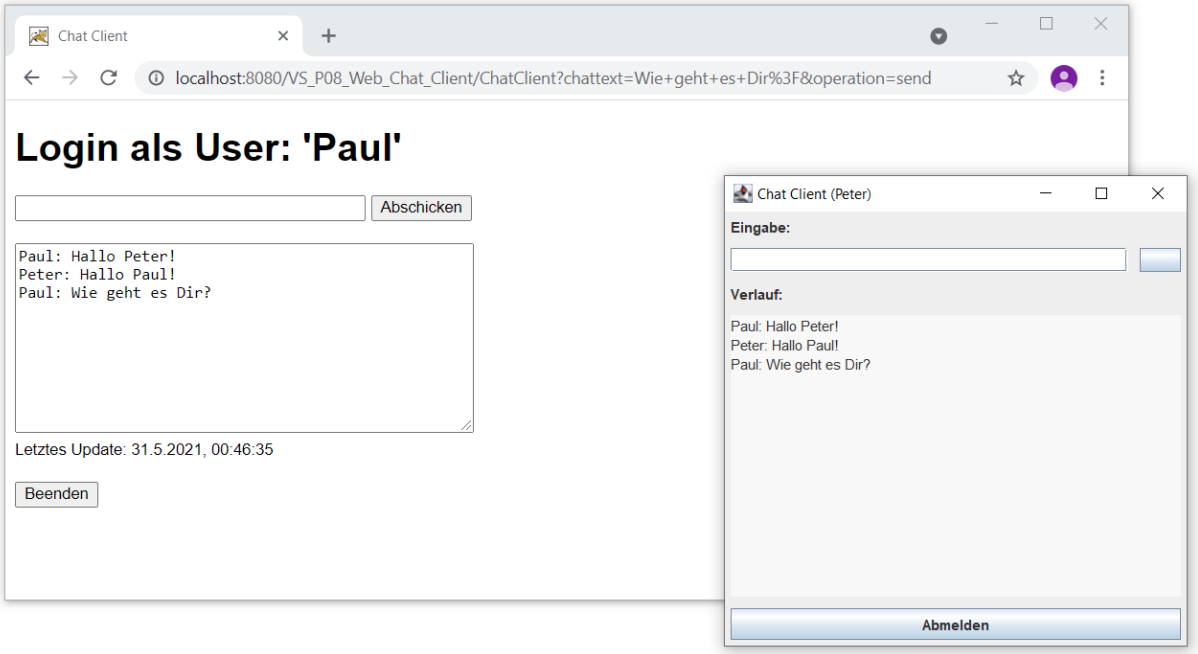
Aufgabenblatt 7

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:

- 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.



- Die Realisierung soll per Java-Servlets sowie statischen Seiten (`html` oder `jsp`) erfolgen.
- Der RMI-Server soll durch das Servlet zum Nachrichten-Austausch genutzt werden. Der native RMI-Client aus dem vorhergehenden Aufgabenblatt soll weiterhin parallel zum Web-Client genutzt werden können.
- 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.
- Erklären Sie im Praktikumsprotokoll den Aufbau Ihrer Lösung anhand eines **Paket-Diagramms**.

Hinweise:

Sie sind recht frei, eigene Wege zur Lösung der Aufgabenstellung einzuschlagen. Damit die Aufgabe auch durch Studierende mit geringeren Vorkenntnissen im Bereich der Web-Programmierung bearbeitet 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. Will man z.B. eine Operation **`init`** im Server mit einem **`GET`**-Request aufrufen, so ist die Nutzung eines **`hidden`-Feldes** in einem Formular sinnvoll:

  ```
  <input type="hidden" name="operation" value="init">
  ```

- Am komplexesten ist die Aktualisierung der Anzeige der empfangenen Chat-Nachrichten. . Für die Ausgabe kann z.B. verwendet werden:
  `<textarea name="chatoutput" cols="50" rows="10"readonly>`
- Sie sollten die nach einem erfolgreichen Login erzeugte **`HttpSession`** mit Verweisen auf die benutzer-spezifischen Chat-Objekte (zumindest die Implementierung der Schnittstelle **`ClientProxy`** zum Empfang von Nachrichten via **`receiveMessage()`**) versehen.
- Da der Client zustandslos ist, müssen die Nachrichten bei einer Aktualisierung komplett via http vom Server zum Client transferiert werden. Aus Effizienzgründen könnte man den Austausch auf **eine Liste der letzten *n* Nachrichten** beschränken.
- Beim *Polling* wird die Anzeige der empfangenen Nachrichten in **zyklischen Intervallen** ständig aktualisiert werden. Die einfachste Möglichkeit dies umsetzen ist durch das kontinuierlichen Absetzen von **`GET`**-Requests die Seite einfach neu zu laden :
  `<meta http-equiv="refresh" content="10;`
  `URL=ChatClient?name=heijo&operation=show">`
  Alternativ, könnte auch ein **`div`**-Container mit den empfangenen Nachrichten per JavaScript aktualisiert werden.
- Achten Sie 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 Web-Client-Instanzen nicht in Form von unterschiedlichen Sessions unterschieden. Beachten Sie, dass unterschiedliche Browser u.U. den selben Cookie-Store verwenden (Chrome und Firefox). Es würde also zu Problemen kommen, wenn man sich in einem Client abmelden würde.
- 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. Das Protokoll zu dieser Aufgabe soll neben der Beschreibung der Umsetzung und der durchgeführten Tests insbesondere ein Paketdiagramm der Lösung beinhalten.

Testierung: 23./24.5.2023