**TW-Mailer Basic Text**

Herzlich Willkommen zum Präsentationsvideo unseres TW-Mailer Beispiels. Ich möchte in diesem Video zuerst auf die Funktionsweise des Programms und anschließend auf einige interessante Code-Stellen eingehen.

Beginnen wir mit dem Server. Hier werden die Eingaben und Befehle des Clients bearbeitet. Beim Starten ist es notwendig den Port, über den sich der Client dann verbindet, sowie ein Mail-Spool Directory anzugeben, in dem die Mails der User persistent abgespeichert werden. Der Server befindet sich nun in einer Endlosschleife und wartet auf eine Verbindung.

Wir können nun den Client starten, indem wir die IP-Adresse des Servers und den vorhin spezifizierten Port angeben. Wenn keine IP-Adresse angegeben wurde verbindet sich der Client automatisch mit der localhost-Adresse. Bei erfolgreicher Verbindung werden wir begrüßt und auch am Server erhalten wir die Nachricht, dass sich ein Client verbunden hat.

Der Client hat nun folgende Befehle zu Verfügung:

**SEND** zeigt dem Server, dass eine Mail geschickt werden will. Es folgt dann auf Client-Seite die Angabe des Senders, des Empfängers, des Betreffs und die Nachricht. Die Nachricht kann beliebig lang sein und über mehrere Zeilen gehen. Gesendet wird das Mail durch das Schreiben einer neuen Zeile mit einem einzelnen Punkt.

[ FILE EXPLORER]

Nun wurde für unseren Empfänger im Mail-Spool Directory ein Verzeichnis erstellt, in dem das File abgespeichert wurde. Wie dies genau funktioniert wird später genauer erklärt.

Bei den Usernamen und beim Betreff gibt es Formatvorgaben bezüglich Länge und Sonderzeichen. Werden diese nicht eingehalten, so reagiert der Server darauf und verwirft die Mail.

Mit **LIST** kann sich der Client alle Mails eines bestimmten Users anzeigen lassen. Es werden hierzu die fortlaufende Mail-ID und der Betreff angezeigt.

Mit **READ** kann sich der Client nun ein bestimmtes Mail aussuchen und sich dessen Inhalt auf der Konsole ausgeben lassen. Wird ein User bzw. eine Mail-ID angegeben, die nicht existiert, so antwortet der Server mit Error.

**DEL** funktioniert sehr ähnlich wie READ, jedoch wird hier das angegebene Mail aus dem Directory gelöscht und kann nicht mehr mit LIST indexiert werden.

Zuletzt gibt es noch den **QUIT** mit dem der Client die Verbindung mit dem Server beendet. Auf Server-Seite kann das Programm nur mit CTRL+C beendet werden.

[ VS CODE]

Als interessantes Code-Beispiel haben wir die Erstellung des Empfänger-Verzeichnisses und des neuen Mails gewählt.

Sobald der Empfänger erfolgreich empfangen wurde wird überprüft, ob es im Mail-Spool bereits einen Ordner für diesen User gibt, wenn nicht wird ein neuer erstellt.

Anschließend wird der Betreff empfangen. Ist dies erfolgreich, so wird am Server das User-Verzeichnis geöffnet und überprüft, welche Mail-ID im Verzeichnis die größte ist. Diese wird dann inkrementiert und anschließend mit einer eindeutigen UUID konkateniert um den Filenamen zu bilden.

Anschließend werden alle empfangenen Daten der Email in das File geschrieben.

Zu Verbessern wäre hierbei, dass die höchste ID immer in einem Index-File gespeichert wird, um die Performance des Programms zu erhöhen, da dadurch das Verzeichnis nicht immer komplett durchsucht werden muss. Weiters ist die UUID überflüssig, da die fortlaufende ID das File auch eindeutig identifizieren kann.

Ev. List Funktionalität

Bei LIST wird wieder das Verzeichnis des angegebenen Users geöffnet und durchlaufen. Für jedes File wird die Zeile des Betreffs, sowie die Mail-ID im Filename zwischengespeichert und ein Counter mitgezählt. Zuerst wird dann die Anzahl der gefundenen Mails an den Client geschickt und anschließend die Auflistung mit ID und Betreff.