Dokumentation Netzwerkprotokoll

Definition

Das Netzwerkprotokoll wird verwendet, um Commands zwischen Client und Server auszutauschen und entsprechend darauf zu reagieren.

Form / Serialisierung

Auf Server- und Clientseite existieren die Packets als Klassen mit unterschiedlichen Inhalten (sog. ContentTypes). Um diese zu übermitteln werden Sie in Strings umgewandelt in folgender Form:

COMMAND_NAME -/- {Inhaltfeld1 -/- Inhaltfeld2} -/- ContentType

-/- Separator: Als Separator wird folgende Zeichenfolge verwendet “-/-”. Es wurde eine selten verwendete Abfolge von Zeichen verwendet, um Fehler bei der Übermittlung zu vermeiden.
COMMAND_NAME: vordefinierte Befehle (siehe Befehle)
{Inhalt}: Der Inhalt wird durch den ContentType vorgegeben. Der Inhalt ist von {} umschlossen und die Felder ebenfalls durch den gewählten Separator unterteilt.
ContentType: gibt Form des Inhalts vor, wird für das Entpacken verwendet.

Befehle

SEND_CHAT	Chatnachrichten an alle Benutzer versenden
NEW_USER	Benutzer mit Benutzernamen erstellen
SET_USERNAME	Anfrage des Clients, den Benutzernamen zu ändern
USERNAME_CONFIRMED	Server bestätigt dem Client die Änderung des Benutzernamens, oder teilt mit, dass der Benutzername bereits vergeben war und deswegen abgeändert wurde.
CHANGED_USERNAME	Server teilt allen Nutzern mit, wenn ein Benutzername geändert wurde
USER_DISCONNECTED	Mitteilung, dass eine Clientverbindung getrennt wurde
PONG PING	Implementierung der Verbindungsüberprüfung. PING von Server zu Client. PONG von Client zu Server
INVALID_ACTION_WARNING	wird gebraucht wenn der User eine invalide Aktion durchzuführen versucht, welche das grundsätzliche Vorgehen nicht stört.
INVALID_ACTION_FATAL	Wird gebraucht wenn der User eine invalide Aktion durchführt und dadurch das Funktionieren des Programms beeinflusst wird.

* Befehle werden erweitert während der Umsetzung des Projektes.

User Commands

Der Benutzer muss mittels CommandLine ebenfalls die Möglichkeit haben, gewisse Änderungen vorzunehmen. Diese werden separat zum Netzwerkprotokoll gehandhabt.

changeusername	Ein Dialog zum Ändern des Benutzernamen wird aufgerufen
quit	Command damit der Benutzer die Applikation beenden kann

*Diese Commands werden während der Umsetzung erweitert.

Implementierung

Im Package ***shared*** implementiert, damit Server und Client beide darauf Zugriff haben.

models: Implementierung der ContentTypes.
- ChatMessage: versenden von Chatnachrichten.
  - String username: Username des Sender
  - Message
- UsernameChange: Wechsel des Benutzernamen.
  - oldName
  - newName

coder: En- und Decoder für die jeweiligen ContentTypes

Commands Implementierungen & Beispiele

SEND_CHAT
Client: Um Chatnachricht an Server zu schicken wird der UserInput in ein Packet mit dem Command SEND_CHAT verpackt.. (CommandLineHandler)
Um erhaltene Nachricht im Terminal anzuzeigen wird ein erhaltenes Packet mit dem Command SEND_CHAT entsprechend dargestellt (CommandLinePresenter)

Server: Um Nachricht von User an alle anderen User zu broadcasten.

NEW_USER
Client: Der Benutzer muss als erstes seinen Benutzernamen wählen und dieser wird mittels dem Befehl NEW_USER an den Server geschickt. (UsernameHandler - chooseAndSendUsername)

Server: Um neuen Benutzer hinzuzufügen und allen anderen Benutzern mitzuteilen, dass jemand beigetreten ist

PING
Server: sendet Ping Nachrichten an alle Client (protocol - pingpong - ServerPingPongHandler - handleConnectedClients())

PONG
Client antwortet auf erhaltene PINGS mit einem PONG an Client. (ClientPingPongHandler - handleTimeForDisconnectOver())

SET_USERNAME
Client: fragt Username-Wechsel beim Server an (siehe User Commands). Packet mit SET_USERNAME wird erstellt. (UsernameHandler - changeAndSendNewUsername())

USER_DISCONNECTED
Server: Wenn ein Client die Verbindung trennt reagiert der Server drauf (ClientCommandHandler), zudem wird ein packet mit diesem Command an Clients gesendet um Sie zu informieren, dass der User ausgetreten ist (UserService - handleUserDisconnected)

Client: erhält Packages mit diesem Command und stellt dies dar (CommandLinePresenter)

CHANGED_USERNAME
Server: Broadcasted den geänderten Usernamen eines Clients an alle anderen Clients. (UserService - updateUsername())

USERNAME_CONFIRMED:
Server: Antwort auf SET_USERNAME des Clients. Bestätigt die Änderung des Benutzernamens und teilt mit, ob der gewählte Nutzername gesetzt werden konnte oder ob dieser abgeändert wurde. (UserService - handleNewUser() und updateUsername())

INVALID_ACTION_WARNING
Server: Antwort auf invaliden Befehl. Wird z.B. ausgelöst wenn der Benutzer den Usernamen zu Laufzeit zu ändern versucht, die clientseitige Validation umgeht und dieser ungültige Benutzername beim Server ankommt (server - services - UserService)

INVALID_ACTION_FATAL
Server: wird ausgelöst wenn der Benutzername invalid ist bei der initialen Wahl des Benutzernamens. Da der Benutzer blockiert ist bis zur Wahl des Benutzernamens wäre dies fatal. → sendet Packet mit errormessage an Client, damit dieser entsprechend reagiert. (siehe UserService - handleNewUser()) und auf Clientseite im ServerResponseHandler und im CommandLinePresenter.