Ziel

Das Ziel der Aufgabe war die Entwicklung eines Clients, bei dem nach Eingabe eines Datums die zu dem Tag gehörenden Temperaturwerte, sowie Minimal-, Maximal- und Durschnittwert auf der Konsole angezeigt werden. Die Daten (24 Werte pro Tag) werden von der RMI-API des ebenfalls zu entwickelnden Servers abgefragt. Der Server soll außerdem eine Möglichkeit bieten Wetterdaten zu ändern. Diese geupdateten Daten sollen dann via RMI-Callback an alle Clients weitergereicht werden. Der Client soll daraufhin die Wetterdaten des kompletten Tages ausgeben und den geupdateten Wettereintrag bei der Ausgabe kenntlich machen.

Protokoll

Format der Anfrage von Client zum Server:

- *Format:* "yyyy-MM-dd "
- *Beispiel:* "2017-10-26" oder "2017-05-01"

Format der Antwort von Server zu Client:

Der Server antwortet mit Temperaturwerte in richtiger Reihenfolge (sortiert nach Uhrzeit von 00:00 nach 23:00):

- *Format:* "TempValue1;TempValue2;...;TempValue24"
- *Beispiel:* "10.2,10.9,...,8.3"
  - *Dezimaltrennzeichen:* '.' (Punkt)
  - *Werttrennzeichen:* ',' (Komma)
  - *keine Angabe der Einheit, es ist von °C auszugehen*

Bei einem Fehler ist die Server-Antwort wie folgt zu implementieren:

- *Fehler-Format:* "ERROR: <Fehlertext>"
- *Beispiel:* "ERROR: Das ist ein Beispiel Fehlertext."

Interfaces

Server:

List<MeasurePoint> getTemperatures(Date date)

boolean register(WeatherClient client)

boolean deregister(WeatherClient client)

Client:

void updateTemperature(MeasurePoint point)

Außerdem wurde sich im Zuge der Übung auf eine Klasse MeasurePoint mit folgenden Attributen geeinigt:

Date _timeStamp;

float _temperature;

Lösung

Client

- Der Client registriert sich initial beim Wetterserver
- Der Client wartet anschließend auf eine Datumseingabe des Users
- Wird ein valides Datum eingegeben, fragt der Client beim Server die gewünschten Daten ab und stellt diese in der Konsole dar

Callback Server

- Erhält der Client von Server ein Callback mit einem aktualisierten Wert, so merkt sich der Client diesen Wert und stellt eine neue Anfrage an der Server mit dem Datum des erhaltenen MeasurePoints
- Die enpfangenen Daten werden dargestellt

Server

Start des Servers:

- Beim Start des Servers wird der RMI-Service Weather an Port 6713 gebunden und gestartet
- Initial werden die Wetterdaten mit Hilfe der Klasse WeatherCSVReader gelesen und als ArrayList gespeichert
- Der Server wartet nun auf Input des Users, dieser kann im folgenden Format einen MeasurePoint updaten:
    - YYYY-mm-DD,h,t
- Bei Eingabe wird der MeasurePoint aktualisiert und an alle Clients geschickt

Anfrage Client

- Bei einer Client-Anfrage werden die entsprechenden MeasurePoints zusammengesucht und an den Client zurück gesendet

WeatherCSVReader

- Einlesen der weather.csv und parsen in eine ArrayList mit Measurepoints