Leo Gruber 3C1 if18b113

Embedded Sensor Cloud – Dokumentation

Benutzerhandbuch:

Der Benutzer verbindet sich mit dem Webserver auf Port 80 (localhost:80) und bekommt daraufhin ein Web Interface zu Verfügung gestellt. Über dieses kann der Benutzer die verschiedenen Plugin Funktionalitäten nutzen und auf der Seite navigieren. Folgende Plugins sind bereits integriert und können wie folgt verwendet werden:

Statische Dateien: Der Benutzer kann einen gewünschten Dateinamen in der Url spezifizieren, woraufhin die gewünschte Datei auf der Weboberfläche dargestellt wird. *Beispiel: http://localhost:80/foo.txt*

Temperatur: Die Seite *htttp://localhost:80/PluginTemperature.html* bietet dem Benutzer ein Interface um Temperaturmesswerte eines bestimmten Tages, oder alle verfügbaren, darstellen zu lassen. Weiters kann der Benutzer mit einer REST Url im Format *htttp://localhost:80/GetTemperature/yyyy/mm/dd* alle Temperaturmesswerte eines bestimmten Tages als XML anfordern.

ToLower: Die Seite *htttp://localhost:80/PluginToLower.html* bietet dem Benutzer ein Interface mit einem Texteingabefeld. In dieses kann der Benutzer einen beliebigen Text eingeben und mittels Buttons in Kleinschreibung umwandeln.

Navigation: Auf der Seite *htttp://localhost:80/PluginNavi.html* hat der Benutzer die Möglichkeit mittels Texteingabefeld nach einem Straßennamen zu suchen, woraufhin alle zugehörigen Orte dargestellt werden. Um diese Funktionalität nutzen zu können muss der Benutzer zuvor den Button „Straße neu aufbereiten“, woraufhin eine Straßen-Ort Zuordnung erstellt wird.

Lösungsbeschreibung:

Zu Beginn war relativ unklar wie die Aufgabenstellung als Ganzes umgesetzt werden kann, beziehungsweise waren die Zusammenhänge nicht leicht erkennbar. Daher wurde mit der Implementierung der Basisklassen (Url, Request, Response, Plugin Manager) begonnen, mit strikter Orientierung an den bestehenden Unit Tests. Genauso wurde die Socket Kommunikation serverseitig implementiert, um Clientanfragen zu ermöglichen und entgegenzunehmen. Als das Grundgerüst stand, sind die Zusammenhänge zwischen den einzelnen Klassen klarer geworden, woraufhin die Aufgabe der Plugin Entwicklung überschaubarer wurde. Jedes Plugin erhält je nach eingehender Request, also je nach Url, eine bestimmte Priorität zugewiesen. Diese entscheidet darüber welches Plugin die eingehende Request behandeln soll. Für die Plugins wurden daher Request Urls gewählt, die eine Zuordnung zu demjenigen Plugin ermöglichen. Darauf aufbauend wurde die Prioritätsbestimmung implementiert. Als nächstes wurde das statische File Plugin implementiert, da es essenziell für das Webinterface, und die Komplexität überschaubar war. Aufbauend auf dem diesem Plugin wurde das Webinterface erstellt worüber die Plugins später bedient/getestet werden können. Die spezifischen Requests wurden mittels JS und Ajax Calls getätigt, die bei bestimmten Nutzer Eingaben getriggert werden. Das ToLower Plugin ist daraufhin als nächstes implementiert worden. Bei diesem wird bei der Texteingabe auf der Weboberfläche durch einen Nutzer ein entsprechender POST Request getriggert, woraufhin das ToLower Plugin die Umwandlung übernimmt. Als Datenbank wurde eine PostgreSQL Datenbank erstellt, genauso eine Klasse die Datenbankverbindungen aufbaut und verwaltet, sowie eine Klasse, die den Zugriff auf die Datenbank

ermöglicht. Darauf aufbauend wurde das Temperatur Plugin implementiert. Im letzten Schritt wurde das Navigation Plugin implementiert. Die Erstellung der Straßen-Ort Zuordnung wurde mittels SAX-Parser realisiert, die Verwaltung dieser übernimmt eine eigene Klasse. Diese Klasse enthält die realisierte Zuordnung und schützt diese bei parallelen Zugriffen (Lock, synchronisierte Methoden), da speziell während dem Parsing-Prozess keine weiteren Zugriffe erlaubt sind.

Reflexion:

Da ich bisher kein Programmierprojekt in diesem Umfang umgesetzt habe, bin ich in erster Linie stolz darauf die Übung, so gut es mir möglich war, umgesetzt zu haben. Der Zeitaufwand war demnach auch relativ groß, jedoch bin ich mit dem Lerneffekt durchaus zufrieden, vor allem da ich zuvor keine Kenntnisse in Java hatte. Genauso bin ich mit meinem Zeitmanagement zufrieden, da ich mir die Arbeit über das Semester aufgeteilt habe und nun keinen Stress gegen Abgabeende habe.

Bei zukünftigen Projekten nehme ich mir vor die Dokumentation der Klassen/Methoden nebenher zu machen und nicht erst am Ende des Projektes. Programmiertechnisch möchte ich in der Zukunft vor allem darauf achten redundanten Code zu verhindern.