Europaschule SZ Utbremen

Cheftrainer Football Manager

Abschlussdokumentation

Bley Robin & Alexander Brennecke

29.02.2016

Inhalt

Einleitung

Der Abschlussbericht des Spiels „Cheftrainer Football Manager“ umfasst im Wesentlichen den Softwaretechnischen Aspekt der Entwicklung und das Projekt Management. Außerdem wird erläutert, in wie weit die gesetzten Ziele des Umsetzungsgrades und der Qualität erreicht wurden. Im Anhang dieses Dokumentes findet sich außerdem eine Liste von verwendeten Software-Komponenten sowie eine Anleitung des Spieles.

Aufbau der Anwendung

Das gesamte Projekt ist in 3 Subprojekte aufgeteilt, die ServerApplication, die ClientApplication und die ConnectorLib. Diese Aufteilung ermöglicht es die ClientApplication und die ServerApplication voneinander unabhängig zu entwickeln. Beide Projekte inkludieren die ConnectorLib. Diese stellt wichtige Komponenten bereit, welche von beiden Projekten benötigt werden. Dazu zählen das Modell, sowie die Kommunikation zwischen der ClientApplication und der ServerApplication. (siehe Visualisierung im Anhang).

ServerApplication

Die Anwendung, welche auf dem Server läuft. Sie kommuniziert mit einer oder mehreren ClientApplications. Zusätzlich verwaltet sie die SQLite Datenbank und regelt den Zugriff auf diese. Der Server dient also ebenfalls als zentraler Speicherort Benutzerbezogener Daten, um diese einem Client ortunabhängig auszuhändigen. Außerdem erledigt sie autonome Aufgaben, wie das Erzeugen der Datenbank, beim ersten Start der Anwendung, anhand der Website www.sportal.de. Dazu wird das HTML der Website mittels Jsoup geparsed. Während des laufenden Betriebs werden weitere autonome Aktionen, wie das „updaten“ der Spielerpunkte, nach einem Spieltag, oder das Ausführen von Transaktionen.

ClientApplication

Die ClientApplication ist im Wesentlichen für die Interaktion mit dem Server zuständig. Außerdem erzeugt sie die GUI, welche mittels Java FX realisiert wurde, mit dessen der Nutzer interagieren kann.

Das eigentliche Spiel ist für den Benutzer die Clientapplikation. Sie ist der Teil, in dem der Nutzer sein Spiel vollzieht. Der Benutzer kann sich also in dieser Registrieren, Anmelden und das Spiel spielen. Des Weiteren kann der Benutzer sich über die Clientapplikation mit verschiedenen Servern verbinden. Jede Veränderung, welche vom Benutzer aus gespeichert werden sollen, wir automatisiert an die Serverapplikation weitergeleitet um diese Zentral zu speichern.

ConnectorLib

Die ConnectorLib beinhaltete das Modell der Anwendung. Zusätzlich bietet sie die Möglichkeit, eine Server/Client Anwendung leicht zu erstellen. Dadurch erhält die ClientApplication einen MessageController und die ServerApplication einen MessageController pro eingeloggten Client. Dem MessageController können Message Objekte übergeben werde. Diese werden automatisch in JSON Formatiert und AES verschlüsselt übertragen. Damit die AES Verschlüsselung korrekt funktioniert führt die ConnectorLib direkt nach der Verbindung einen Handshake zwischen Client und Server durch. Dabei wird ein AES Key erzeugt und RSA verschlüsselt zwischen Client und Server übertragen.

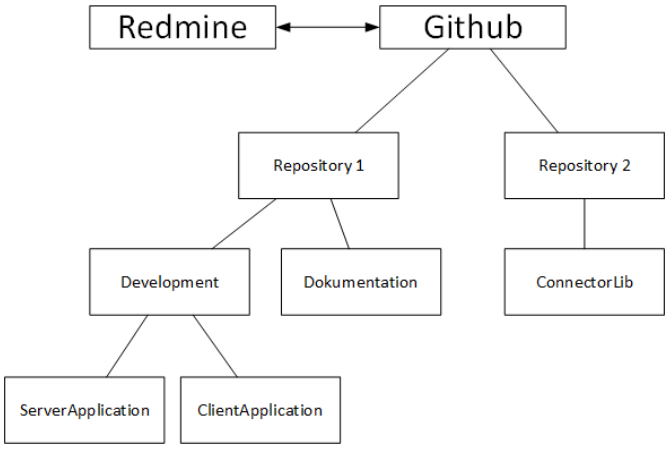
Ein Message Objekt besteht im Wesentlichen aus einer ID und der eigentlichen Nachricht, welche als String übergeben werden kann. Die ConnectorLib stellt zwei Klassen, eine für „Client-To-Server“ und

eine für „Server-To-Client“ Message IDs zur Verfügung. Die dort registrierten IDs können einer Message im Konstruktor übergeben werden. Bei der Erstellung des Message Controllers werden diese IDs auf Klassen gemapped. Dafür wird ein Classloader verwendet, welcher Instanzen der Klassen erzeugt und diese in einer Map speichert. Diese Klassen müssen in einem „callables“ Package liegen, der Name der Klasse muss äquivalent zu dem Wert der ID sein und sie müssen von CallableAbstract erben. Dies hat den Vorteil, dass kein Switch-Case verwendet werden muss. Eine Nachricht wird direkt „messageArrived (Message message)“ der Klasse mit der entsprechenden ID weitergeleitet.

Sowohl dem Server als auch dem Client stehen eine Reihe von Message Templates zur Verfügung. Diese können mit Objekten initialisiert und direkt gesendet werden. Die Umwandlung in einen validen JSON String übernimmt dabei das gewählt Message Template. Es ist jedoch auch möglich einen einfachen JSON String zu übermitteln. Dafür kann die „Message“ Klasse verwendet werden, welche lediglich den Inhalt sowie eine der „Client-To-Server“ oder „Server-To-Client“ Message IDs benötigt. Auch diese Nachricht wird verschlüsselt übertragen.

Projekt Management

Während der Arbeit an dem Projekt wurde mit dem Ticketsystem Redmine gearbeitet. Der Entwickelte Code wurde in zwei Repositorien aufgeteilt und auf GitHub zur Verfügung gestellt. Daher war es möglich an verschiedenen Orten und Rechnern an dem Projekt zu arbeiten. Ein genauer Aufbau ist der nebenstehenden Grafik zu entnehmen. Eine Verknüpfung zwischen Redmine und GitHub hat die Vorteile, dass Zeiten, Fortschritt und Kommentare direkt in einer Commit Message an das entsprechende Ticket geschrieben werden können. Dadurch war es zu jeder Zeit möglich den Fortschritt des Projektes zu überwachen.



Beschreibung des Umsetzungsgrades

Die Kriterien der Anforderungen der Applikation wurden größten Teils umgesetzt. Dabei wurden fast alle Musskriterien und Wunschkriterien erfüllt. Um diese Kriterien und Aufgaben im Quellcode wiederzufinden wurde versucht entsprechende Funktionen mit den Tags zu markieren. Leider kann dies nicht immer gelingen, da Quellcode nicht immer genau einem Kriterium zugeordnet werden kann.

Folgende Musskriterien wurden umgesetzt:

- Nutzer können sich Registrieren (**/F0011/**)
- Nutzer können sich Anmelden und Abmelden (**/F0020/**)
- Nutzer können Spielrunden Gründen und beitreten (**/F0012/**)
- Nutzer können jederzeit mit einem Spielrundennamen und dem dazugehörigen Passwort einer Spielrunde beitreten (**/F0040/**)

- Nutzer haben Zugang zu einer Spielrundenübersicht, in der sie zwischen ihren Spielrunden wechseln können (**/F0060/**)
- Nutzer können ihre Aufstellung verwalten (**/F0100/**)
- Nutzer können sich den Spielstand anschauen (**/F0310/**)
- Die Anwendung berechnet nach einem Spieltag die Punkte der Spieler und Manager (**/L0030/ und /L0040/**)
- Die Oberfläche ist leicht verständlich und deutschsprachig

Da während der Umsetzung des Projekts der Entschluss gefasst wurde, dass es keine Administratorfunktionen geben wird, wurden einige Kriterien nicht erfüllt. Dabei wurden folgende Musskriterien nicht erfüllt:

- Ein Admin kann Daten einer Spielrunde verändern (**/1140/**)
- Ein Admin kann Strafen und Gutschriften verteilen (**/F1110/**)
- Ein Admin kann Spielrunden löschen **/F1150/**

Diese Wunschkriterien wurden nicht erfüllt:

- Ein Admin kann Nutzer aus einer Spielrunde entfernen (**/F1010/**)
- Ein Admin kann Punkte anpassen (**/F1120/ und /F1130/**)

Jedoch wurden alle Kriterien erfüllt, welche keinen Bezug zu der Implementierung von Administratorfunktionen haben. Folgende Wunschkriterien wurden erfüllt:

- Nutzer können Spieler auf den Transfermarkt setzten (**/F0150/)**
- Nutzer können auf Spieler bieten **/F0230/**
- Nutzer können Gebote einsehen und annehmen (**/F0260/**)
- Der Marktwert eines Spielers wird dynamisch berechnet (**/L0020/**)
- Der Nutzer kann sich seine lokalen Statistiken anschauen (**/F0300/**)
- Das System reagiert dynamisch auf einen Spielerwechsel in der Bundesliga

Zusätzlich wurden alle vorher definierten Produktleistungen abgedeckt:

- Bei einer falschen Eingabe des Nutzers wird dieser darüber informiert. (**/L0010/**)
- Die Marktwertberechnung wurde anders umgesetzt, als zu Beginn des Projektes beschrieben. Dies geschah, da sich herausgestellt hat, dass das vorgeschlagene Verfahren zu gradlinige Werte liefert. Es wurde ein Algorithmus entworfen, welcher mehrere Faktoren in die Berechnung einfließen lässt und so sehr zufriedenstellende Marktwerte liefert. (**/L0020/**)
- Die Berechnung der Punkte eines Spielers und eines Managers funktioniert exakt so, wie es festgelegt wurde. (**/L0030/ und /L0040/**)
- Die Spielerdatenbank wird automatisch bei dem ersten Serverstart generiert. Jedoch beziehen wir die Spielerinformationen auch von www.sportal.de und nicht wie vorher festgelegt von ran.de. Diese Umstellung ermöglicht es, die Spieler bei der Punktevergabe leichter zu erkennen, da Sportal diese mit IDs versieht. Zudem ist es momentan nicht implementiert, dass Spieler aus der Datenbank verschwinden, oder hinzugefügt werden. Dies liegt daran, dass wir keine zufriedenstellende Möglichkeit gefunden haben diese Spielerwechsel zu bemerken. Die einzige Möglichkeit, welche wir gefunden haben, beruht darauf, immer wieder die Datenbank mit der Sportal Website zu vergleichen. Da dieses

Verfahren jedoch sehr viele Ressourcen benötigt haben wir davon vorerst abgesehen. (**/L0050/**)

- Die Geldvergabe wurde so umgesetzt, wie es festgelegt wurde. (**/L0060/**)

In der nachfolgenden Tabelle findet sich eine Übersicht über alle aufgestellten Produktfunktionen. Einige der Produktfunktionen wurden zu Beginn des Projektes nicht in die Kategorie Wunsch- oder Musskriterium eingeteilt. Diese Kriterien wurden teilweiße umgesetzt, teilweiße jedoch auch Ressourcenmangel vorerst nicht umgesetzt.

Gruppe	Kriterium	Umgesetzt
Benutzerfunktionen	/F0010/ Registration	
	/F0011/ Nutzer Registration	
	/F0012/ Spielrunden Registration	
	/F0020/ Anmeldung	
	/F0030/Abmeldung	
	/F0040/ Spielrunde beitreten	
	/F0050/ Spielrunde verlassen	
	/F0060/ Spielrundenübersicht	
	/F0061/Spielrunde auswählen	
	/F0100/ Aufstellung verwalten	
	/F0110/ Spielerübersicht	
	/F0120/ Spieler zur Aufstellung hinzufügen	
	/F0130/ Spieler aus Aufstellung entfernen	
	/F0140/ Formation der Aufstellung ändern	
	/F0150/ Spieler verkaufen	
	/F0200/ Transfermarkt	
	/F0210/ Transfermarktübersicht	
	/F0220/ Transfermarkthistorie	
	/F0230/ Angebot für Spieler erstellen	
	/F0240/ Angebot für Spieler löschen	
	/F0250/ Gebotsübersicht	
	/F0260/ Gebot annehmen	
	/F0300/ Statistiken	
	/F0310/ Spielstand anschauen	
	/F0320/ Punkte des letzten Spieltages anschauen	
	/F0330/ Tabellenplatzstatistik	
	/F0340/ Punkte eines Spielers anschauen	
Administratorfunktionen	/F1000/ - /F1130/	
Produktleistungen	/L0010/ Akkumulation	
	/L0020/ Marktwertberechnung	
	/L0030/ Punktevergabe	
	/L0040/ Nutzerpunkte berechnen	
	/L0050/ Automatische Aktualisierung der Spielerdatenbank	
	/L0060/ Geldvergabe	

Testabdeckung

Die, im Pflichtenheft definierten, Testfälle wurden nur teilweiße umgesetzt. Dafür gibt es drei verschiedene Gründe. Die Testfälle /T0110/, /T0130/, /T0211/, /T0221/, /T0222/, /T0223/ wurden nicht umgesetzt, da sie als reine GUI-Tests definiert wurden. Zwar ist es generell möglich, eine mit JavaFX erstellte GUI zu testen, jedoch hätte diese Umsetzung den zeitlichen Rahmen des Projektes bei weitem überschritten. Der Testfall /T0030/ wurde nicht umgesetzt, da es nicht möglich ist dieses Verhalten sinnvoll zu testen. Außerdem wurde die Testfälle, welche die Administratorfunktionen testen sollten, nicht umgesetzt, da die Administratorfunktionen selber nicht umgesetzt wurden.

Um die Tests konstant zu überwachen wurde das Tool Travis-CI (https://travis-ci.com/) verwendet. Dies bietet die Möglichkeit bei Codeveränderungen automatisiert zu testen. Dadurch kann sichergestellt werden, dass eine Codeveränderung keine Fehler in bereits existierenden Code auslöst. Leider war es nach der Implementierung der datenbankspezifischen Tests nicht mehr möglich, dieses Tool zu benutzen.

Die genaue Testabdeckung ist der nachfolgenden Tabelle zu entnehmen:

Testfall	Umgesetzt
/T0010/ Nutzerregistrierung	Ja
/T0020/ Anmeldung	Ja
/T0030/ Abmeldung	Nein
/T0100/ Spielrunde beitreten	Teilweise
/T0110/ Spielrundenübersicht	Nein
/T0120/ Spielrunden Registration	Ja
/T0130/ Spielrunde auswählen	Nein
/T0200/ Spielen	Teilweise
/T0210/ Transfermarkt	Teilweise
/T0211/ Transfermarkt anschauen	Nein
/T0212/ Gebot abgeben	Ja
/T0213/ Gebot löschen	Ja
/T0214/ Gebot annehmen	Ja
/T0220/ Aufstellung	Teilweise
/T0221/ Aufstellung anschauen	Nein
/T0222/ Formation ändern	Nein
/T0223/ Spieler wechseln	Nein
/T0224/ Spieler verkaufen	Ja
/T1000/ Ändern der Spielrundendaten	Nein
/T1010/ Gutschrift erteilen	Nein
/T1020/ Strafe erteilen	Nein
/T1030/ Spielrunde löschen	Nein
/T2000/ Marktwertberechnung	Ja
/T2010/ Parser	Ja
/T2011/ Spielplan	Ja
/T2012/ Noten	Ja
/T2013/ Spieler	Ja
/T2020/ Team Generierung	Ja
/T2030/ Punkteberechnung	Ja
/T2040/ Nutzerverwaltung	Ja (wurde in /T0020/ getestet)

Analyse der Qualitätszielbestimmungen

Zu Beginn des Projektes wurden Qualitätsbestimmungen aufgestellt. Diese wollen wir nun reflektieren und analysieren. Die nachfolgende Tabelle zeigt, in wie weit wir die Umsetzung in diesen Punkten bewerten. Dabei repräsentieren die schwarzen Kreuze das aufgestellte Ziel und die roten/grünen Kreuze den heutigen Umsetzungsgrat.

	SEHR WICHTIG	EHER WICHTIG	NICHT WICHTIG
ROBUSTHEIT		XX	
ZUVERLÄSSIGKEIT	X	X	
KORREKTHEIT		X X	
BENUTZERFREUNDLICHKEIT	X	X	
EFFIZIENZ		X	X
PORTIERBARKEIT		XX	
KOMPATIBILITÄT			XX

Bei der Zuverlässigkeit und der Benutzerfreundlichkeit sind leichte Mängel zu begutachten. Dies lässt sich über die Abhängigkeit zu Sportal.de begründen. Da Sportal.de keine API bereitstellt muss das HTML der Seite geparsed werden. Leider baut sportal.de dort teilweiße nicht funktionierende Links ein. Daher kommt es teilweiße vor, dass einzelne Spiele nicht erfasst werden. Daher bewerten wir die Zuverlässigkeit schlechter. Die Benutzerfreundlichkeit bewerten wir schlechter, da die GUI an manchen Stellen etwas unübersichtlich ist. Diese Mängel sind in unseren Augen nicht gravierend, jedoch hätte die Gestaltung an manchen Stellen besser sein können.

Erkenntnisgewinn

Ein Erkenntnisgewinn fand größtenteils in drei Bereichen statt: Dem erweiterten Umgang mit Maven und speziell dem Dependency Management, der Umsetzung einer grafischen Oberfläche mittels JavaFX sowie der Automatisierung der ServerApplication. Alle drei Bereiche haben uns immer wieder vor Herausforderungen gestellt, welche oft erst nach einigen Versuchen zu meistern waren. Jedoch haben diese Herausforderungen dazu beigetragen, dass sich unsere Kenntnisse über alle drei Bereiche hinweg deutlich verbessert haben. Zusätzlich war es bereichernd in einem größeren Projekt auf die Struktur der Anwendung und des Codes zu achten und dabei altes, beinahe vergessenes Wissen, beispielsweiße über SQL, wiederaufzufrischen. Darüber hinaus gewannen wir neue Kenntnisse im Bereich des Datenaustauschs zwischen mehreren Teilnehmen über ein Netzwerk mittels Sockets in Java. Dabei lernten wir vor allem die Verarbeitung von empfangenen Daten. Die Besonderheiten einer Serveranwendung wie beispielsweise zeitgesteuerte Aktionen und das Synchronisieren von Clientspezifischen Daten sind aus ebenfalls nähergekommen.

Neben technischen Neuerkenntnissen gewannen wir weitere Erfahrung im Bereich der Planung und der Zeiteinhaltung. Eine präzise Erfassung der Anforderungen und Aufgaben eines Projekts sind ebenso wichtig wie eine realistische Zeitplanung. Unsere Fähigkeiten in der Planung dessen haben sich, wie sich der Vergleich zu vergangenen Projekten zeigt, deutlich verbessert.

Anhang

Anleitung

Registrierung und Anmeldung

Für die Registrierung muss im Startmenü „Registrieren“ gewählt werden. Daraufhin öffnet sich ein Menü für die Registrierung. Dieses Menü muss mit Daten befüllt werden. Wenn die Server Details korrekt sind, muss für den Abschluss der Registrierung der Button mit der Aufschrift „Registrieren“ gewählt werden. Werden die eingegebenen Daten vom Server akzeptiert wird sich ein Fenster öffnen in dem dies steht. Falls die Server Details oder Eingaben falsch sind, öffnet sich ebenfalls ein Fenster in dem der jeweilige Fehler steht.

Zum Anmelden mit einem bereits angelegten Account, müssen im Startmenü die jeweiligen benötigten Daten des Accounts eingegeben werden. Anschließend muss der Button mit der Aufschrift „Anmelden“ gewählt werden, woraufhin sich entweder eine Fehlermeldung über den Misserfolg der Anmeldung oder das Spiel, bei Erfolg der Anmeldung, öffnet.

Falls die bereits Voreingestellten Server Details fehlerhaft sind, müssen diese Angepasst werden. Dafür muss im Startmenü ein Haken bei der Checkbox „Server Details“ gesetzt werden. Das sich daraufhin öffnende Menü muss mit einer passenden IP-Adresse und einen passenden Port befüllt werden.

Verwaltung von Ligen (Erstellen, Beitreten und Spielen)

Seine Ligen zu verwalten muss der Menüpunkt Communities in dem „Side-Menu“ ausgewählt werden. Für die Erstellung einer Community muss der Button „Community erstellen“ ausgewählt werden. Daraufhin öffnet sich ein Fenster, in dem man den Namen und das Passwort einer neuen Community setzten kann. Nach der Bestätigung der eigegebenen Daten gibt die Anwendung eine Rückmeldung ob die Community erstellt werden konnte oder ein Fehler mit der Verbindung zum Server oder den eingegebenen Daten gibt.

Um einer bereits existierenden Community beizutreten muss der Button „Community beitreten“ angeklickt werden. Anschließend öffnet sich ein Fenster, welches den Namen und das Passwort der Community fordert. Wenn korrekte Daten eingegeben werden und diese bestätigt sind, öffnet sich eine Rückmeldung über den Erfolg oder Misserfolg der Erstellung der Community.

Um in einer Community zu spielen muss eine Community angeklickt werden. Diese Community wird daraufhin markiert. Sie ist solange die aktuelle Community bis eine andere ausgewählt wird. Wenn ein anderer Menüpunkt des „Side-Menus“ ausgewählt wird, wird in dieser Community gespielt.

Handel mit Spielern

Der Transfermarkt für Spieler befindet sich unter dem Menüpunkt „Transfermarkt“ in dem „Side-Menu“. Auf dem Transfermarkt kann man Spieler Anbieter, auf Spieler bieten, Angebote annehmen und ablehnen und Spieler vom Transfermarkt nehmen.

Einen Spieler setzt man auf dem Transfermarkt zum Verkauf, in dem man den Button „Spieler hinzufügen“ anklickt. Es öffnet sich ein Fenster, in dem sich alle eigenen Spieler befinden. Um einen Spieler nun auf den Markt zu setzten muss der Button „hinzufügen“ neben den jeweiligen Spieler geklickt werden. Falls der Spieler, wieder vom Markt genommen werden soll, kann der Button erneut geklickt werden, welcher nun die Beschriftung „entfernen“ enthält.

Um die Angebote auf eigene Spieler und Gebote auf andere Spieler einzusehen, kann der Button „Angebote“ geklickt werden. Anschließend öffnet sich ein Fenster, welches alle Angebote enthält. Dort kann man Angebote annehmen, ablehnen und zurückziehen.

Um auf bereits zum Verkauf stehende Spieler zu bieten, kann ein Spieler mit einem Doppelklick ausgewählt werden, woraufhin sich ein Fenster öffnet, in dem man ein Gebot abgeben kann.

Managen der Aufstellung

Den Zugriff auf seine Aufstellung erhält man unter den Menüpunkt „Aufstellung“ des „Side-Menus“. Dort können Spieler durch das jeweilige anklicken eines Spielers ausgetauscht werden. Mit einem Klick auf einem Spieler, öffnet sich ein Fenster, in dem sich Spieler befinden, welche noch nicht aufgestellt sind und auf der gleichen Position wie der angeklickte Spieler spielen. Die Formation der Aufstellung ändert man in dem man den Button „Formation“ anklickt. Anschließend kann man eine der verfügbaren Formation auswählen. Um vorgenommene Veränderungen an der Formation zu speichern muss der Button „speichern“ ausgewählt werden.

Spielstand prüfen

Um Statistiken über eine eigene Community einzusehen kann der Menüpunkt „Statistiken“ im „Side-Menu“ ausgewählt werden. Dort kann man entweder die aktuellen Punkte aller Manager in der Community oder den Verlauf der eigenen Punkte pro Spieltag einsehen. Diese Statistiken sind über die Buttons „Verlauf“ und „Ranking“ abrufbar.

Verwendete Komponenten

Java FX

JavaFX ist ein plattformübergreifendes Framework zur Erstellung von graphischen Oberflächen. Wir verwendeten JavaFX für die Nutzeroberfläche der ClientApplication. Dieses Framework wurde verwendet, weil es unsere Meinung nach eine schöne Alternative zu Swing ist. Der Hauptgrund für den Einsatz von JavaFX ist jedoch die Tatsache, dass wir uns die Handhabung dieses Frameworks aneignen wollten.

Log4J

Zum Loggen der Kommunikation zwischen Server und Client haben wir das Framework Log4j verwendet. Log4j ist Betriebssystemunabhängig, da es ausschließlich in Java zum Einsatz kommt. Log4j bietet die Möglichkeit an jeder Stelle des Codes einfach in ein Logfile zu schreiben. Zusätzlich sichert Log4j, dass das Logfile nicht unendlich weiterwächst, da Log4j mit einer maximalen Dateigröße konfiguriert werden kann.

Maven

Das Build-Management-Tool Maven wurde in allen drei Subprojekten verwendet. Maven ist ein Tool, welches für den Erstellungsprozess einer Anwendung zuständig ist. Die einfache Einbindung von Bibliotheken war ausschlaggebend für den Einsatz in diesem Projekt. Durch dieses Dependency Management war es möglich, die ConnectorLib Plattform- und DIE-unabhängig in die ServerApplication und ClientApplication einzubinden. Die ConnectorLib, welche in einem eigenen Git-Repository liegt, kann direkt dorthin deployed werden.

SQLite

SQLite ist eine Bibliothek, welche Datenbankfunktionen mit sich bringt. Diese Bibliothek benötigt keine weiteren Serverbedingungen, benötigt einen kleinen Speicherbereich und verwendet eine Art reduzierte SQL database engine. Durch diese Eigenschaften ist SQLite gut für das Projekt Cheftrainer geeignet, da das besagte Projekt ein vergleichsweise wenig umfangreiches ist und nur die grundlegendsten Datenbankfunktionen benötigt. Sie kommt ausschließlich in der ServerApplication zu Einsatz.

JUnit

Für das automatisierte Testen der drei Subprojekte kam das Framework JUnit zum Einsatz. JUnit vereinfacht es Fehler während der Entwicklung zu finden oder gar zu vermeiden, da es mit jedem Erstellungsprozess der Anwendung besagte Tests ausführt und Rückmeldung über den Erfolg dieser liefert.

JDOM

JDOM ist eine speziell für Java entwickelte Bibliothek, welche es einem ermöglicht XML-Dateien zu bearbeiten und zu lesen. Es wird in der ClientApplication benötigt um ein Menü in Abhängigkeit von einer XML-Datei zu generieren.

JSON

Zum Datenaustausch zwischen Client und Server kommt JSON in diesem Projekt zum Einsatz. JSON ist ein einfach lesbares, einfach zu verendendes und kompaktes Datenformat. Diese Eigenschaften, gepaart mit der guten Erfahrung aus mehreren Projekten, sind der Grund für die Verwendung dieses Datenformats.

Jsoup

Das Parsen von Websites, für die Automatisierung des Servers, wurde mithilfe von Jsoup realisiert. Jsoup ist eine Bibliothek, welche das Lesen und Bearbeiten von HTML-Dateien ermöglicht. Es ist deutlich komfortabler und einfacher HTML mittels Jsoup zu parsen als auf einen reinen XML Parser zurückzugreifen.

Aufbau der Anwendung