Fachhochschule Brandenburg
Fachbereich Informatik

Twitchess Dokumenation

im Studienfach Software Qualität

Thema:	Twitchess - Der Twitter Schachbot
eingereicht von:	CurtisMosters<mosters@fh-brandenburg.de>
	Sebastian Minke <minkes@fh-brandenburg.de>
	Benjamin Hoffmann <hoffmanb@fh-brandenburg.de>
	MarcoKnietzsch <knietzsc@fh-brandenburg.de>
eingereicht am:	10. Januar 2012
Betreuer:	Mark Rambow

Inhaltsverzeichnis

Kapitel 1

Einführung

1.1 Vorwort

In dieser Dokumentation schildern wir welche Funktionen [2]Twitchess bietet und wie sich dieser Bot sich benutzen lässt.

Auf dem [1]Jenkins Server lässt sich der aktuelle Status einsehen inklusive der Testabdeckung, welche bei Projektende etwa bei 88% lag.

Ein besonderes Dankeschön geht an Mark Rambow für die gute Betreuung.

1.2 Akronym Twitchess

[2]Twitchess ist wie der Name schon verrät eine Mischung aus Twitter und Chess, zu Deutsch Schach.

Unsere Aufgabe war es einen [2]Twitterbot zu entwickeln, welcher es erlaubt Schach gegen ihn zu spielen.

1.3 Anforderungen

1.3.1 Mindestanforderungen

Im Großen und Ganzen kamen folgende Punkte

- Spielen gegen KI (Anbindung einer Schachengine)
- Anbindung an Twitter (Steuerung über Nachrichten)
- Aktuelle Status-/Spielfeldanzeige durch Grafiken (über Twitter)
- Speicherung laufender und vergangener Spiele/Spieler

in unsere Mindestanforderung, die wir unbedingt umsetzen wollten.

1.3.2 Zusatzanforderungen

Zusätzlich zu den Mindestanforderungen dachten wir natürlich über spezielle Features nach, die bei entsprechender Zeit zu integrieren wären. Folgende Punkte

- Änderung des Schwierigkeitsgrades
- Eigene KI
- Spieler gegen Spieler
- GUI
- Andere Spiele wie TicTacToe oder Vier Gewinnt

wurden aus Zeitmangel daher nur teilweise umgesetzt.

Kapitel 2

Anforderungen und Konfiguration

2.1 Systemanforderungen

Auf dem Zielsystem muss [4]JAVA JRE Mindestversion 1.6.26 [1] installiert sein. Sollte Ihr System nicht über [4]JAVA verfügen können Sie es hier kostenlos herunterladen.

2.2 Konfigurationsmöglichkeiten

Die Einstellungen erfolgen in der Properties Datei "configuration.properties"

- Pfad der Schachengine unter Linux:
  - "ChessEngineFactory.Linux=chessengines/stockfish-211-32-ja-linux"
- Pfad der Schachengine unter Mac:
  - "ChessEngineFactory.Mac=chessengines/stockfish-211-32-mac"
- Pfad der Schachengine unter Windows (wird standardmäßig auch verwendet, wenn das Betriebssystem nicht korrekt erkannt wird):
  - "ChessEngineFactory.Windows=chessengines/stockfish-211-32-ja-windows.exe"
- Twitteraccountname, der standardmäßig beim Starten verwendet wird:
  - "Twitter.StandardAccount=Twitchess1"

[1] Sollte aber auch mit JAVA 7 funktioneren

- Zeit, die der Schachengine für die Berechnung eines Zuges maximal zur Verfügung steht (in ms). Je größer die zur Verfügung stehende Zeit, desto stärker spielt die Schachengine.
  - "Engine.TimePerMove=2000"
- Regeln für das Akzeptieren eines Unentschiedens: Wenn die Bewertung der Schachengine kleiner ist als der Wert(ein Bauer mehr/weniger macht einen Unterschied von 100, eine Dame mehr/weniger macht einen Unterschied von 900)
  - "AcceptDraw.AbsScoreLessThan=100"
- Anzahl der ganzen Züge (d.h. Weiß und Schwarz ziehen in einem ganzen Zug) bevor ein Unentschieden überhaupt erst in Frage kommt
  - "AcceptDraw.FullMoveCountGreaterThan=10"

Kapitel 3

Kommandos und Limitierung

3.1 Befehlskommandos

Ein kleines Beispiel zum Einstieg

- Der Spieler macht einen Zug, dann der Computer[1] [2] [3]
  - "@Name chess move e2e4"
- Der Spieler macht einen Zug, der Computer zieht nicht
  - "@Name chess move player e2e4"
- Spieler lässt einen Zug von Twitchess machen
  - "@Name chess move ai"

Zugarten

- Beispielzüge:
  - "e2e4"
  - "e3d4"
  - "g1f3"
- Rochade:
  - "e1g1"
  - "e1c1"
  - "e8g8"
  - "e8c8"

[1] Mit Name ist der Token gemeint, welcher standardmäßig auf Twitchess1 gesetzt ist
[2] Beispiel: @Twitchess1 chess new
[3] Nachfolgend immer als @Name bezeichnet

- Umwandlung von Bauern in eine/n:
  - Dame: “e7e8q”
  - Turm: “e7e8r”
  - Läufer: “e7e8b”
  - Springer: “e7e8n”
- print
  - Wichtig um zu sehen wie das Spielfeld aktuell aussieht.
  - Befehl: “@Name chess print”

Kontrollbefehle

- new
  - Hiermit kann man ein neues Spiel starten. Falls bereits ein existierendes Spiel vorhanden war bekommt man einen Hinweis.
  - Befehl: “@Name chess new”
- offerdraw
  - Nutzer bietet Remis(Unentschieden) an.
  - Befehl: “@Name chess offerdraw”
- cancel
  - Gut um ein laufendes Spiel abzubrechen.
  - Befehl: “@Name chess cancel”

3.2 Limitierungen

Auf Grund von Twitter müssen wir Sie über folgende Beschränkungen hinweisen auf welche wir leider keinen Einfluss haben. Zum Einen ist es das Nachrichtenlimit, welches theoretisch ziemlich schnell erschöpft ist. Da wir nach jedem Zug ein Bild generieren und es bei Twitter hochladen entstehen durch jeden Zug zwei Statusupdates. Außerdem kann es sein, dass wenn Sie versuchen eine Nachricht zu senden, eine Meldung auftritt, die Ihnen sagt, dass Sie bereits dasselbe getwittert hätten. Dies können Sie umgehen, indem Sie zwischen den einzelnen Parametern der Befehle weitere Leerzeichen einfügen (z.B. statt ”chess newßchreiben Sie ”chess new”.

Kapitel 4

Spielgeschehen

4.1 Spielstart

4.1.1 Installation

Aus Useablity Gründen haben wir bereits für Windows und Unix Systeme fertige Scripte angelegt, welche Sie nutzen können. Durch diese Scripte lässt sich Twitchess nun ohne Probleme starten.

Diese Skripte befinden sich im Hauptverzeichnis und sind nach den System benannt. Bei Windows Systemen lässt sich die Batch sehr bequem ausführen.

- "Windows.bat"

Bei Unix Systemen lässt sich das bereits ausführbar gemachte Shell Skript ausführen.

- "Unix.sh"

Der erste Schritt ist es den Bot zu erstellen

- Befehl: "ant jar"

Folgende Unterordner/Dateien müssen im Verzeichnis der Jar Datei vorliegen:

- Ordner: "img/"
- Ordner: "chessengines/"
- Datei: "configuration.properties"
- Datei: wahlweise den Accesstoken für den Twitteraccount
  - Beispiel: "Twitchess1"

4.1.2 Bot starten

Aufruf der Skripte oder alternativ “java -jar twitchess.jar”. Achten Sie darauf, dass die von Ihnen benutzte Schachengine (z.B. “chessengines/stockfish-211-32-ja-linux”) ausführbar ist.

4.1.3 Verbindung mit Twitter

Um Twitchess mit dem eigenen Twitteraccount zu verbinden, geben Sie am Anfang einfach deinen Benutzernamen ein. Danach erhalten Sie die Nachricht, welche besagt, dass noch kein Token zu deinem Account existiert. Wähle Sie nun “neues Token erstellen”. Kopieren Sie den erscheinenden Link in deinen Browser und geben Sie somit Twitchess Zugriff auf deinen Account. Geben Sie daraufhin den auf der nächsten Seite erscheinenden Pin in die Konsole ein. Jetzt sind Sie mit Ihrem Twitter Account eingeloggt. Ab dem nächsten Start reicht es deinen Twitternamen einzugeben und Sie werden verbunden.

Twitchess entfernen

Um Ihren Account wieder für Twitchess zu sperren gehen Sie in Ihrem Twitteraccount auf “Einstellungen” → “Applikationen” → “Name des Twitteraccounts” → “Zugriff widerrufen”.

Hinweis: Den Standard-Twitter-Account können Sie in der “configuration.properties” Datei ändern. (“Twitter.StandardAccount=Name”)

4.2 Spielablauf

Hier wird Ihnen nun ein exemplarischer Spielablauf aufgezeigt.
Am Anfang muss natürlich erst einmal ein neues Spiel erstellt werden

- Befehl: “@Name chess new”

Wenn alles geklappt hat, erhalten Sie eine Bestätigung. Danach können Züge wie folgt durchgeführt werden. Spieler macht Zug

- Befehl: “@Name chess move e2e4”

Als Antwort bekommt man den Zug von [2]Twitchess und einen Link zum Bild der Stellung.[1]

- Antwort: “@Spieler e7e5 Link zum Bild” [2]

Die Aktuelle Stellung als Bild anzeigen.

[1] Mit Spieler ist der Account von Twitter gemeint, welcher mit dem Bot gerade spielt

[2] Der Link(URL) sollte in der Regel von einem sogenannten Linkshortener kommen und dann weiterverlinken auf die Twitter Fotoseite.

- Befehl: “@Name chess print”
  - Beispielbild:

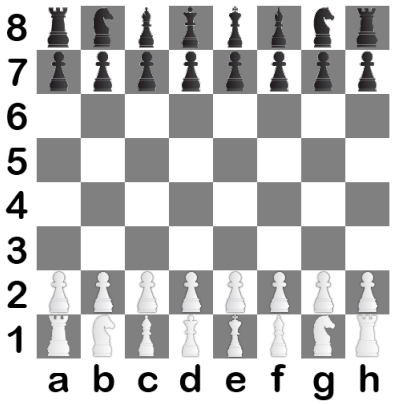

Abbildung 4.1: Beispielbild nach dem print Befehl zu Beginn

4.3 Plattformen

4.3.1 Unterstützte Plattformen

Folgende Betriebssysteme werden unterstützt.

- Windows XP, CE, ME, 2003, Vista, Seven
- Linux sowie sämtliche Unix Distributionen
- SunOS/Solaris
- Mac OS/Mac OS X

4.3.2 Getestete Plattformen

Selbst getestet wurde das Programm unter folgenden Betriebssystemen

- Windows XP
- Windows Seven
- Ubuntu 10.10

Dabei wurde großer Wert auf die möglichst einfache Ausführbarkeit, sowie Spielablauf und -verlauf gelegt. Abstürze gab es bei unseren Tests keine.

Kapitel 5

Verwendete Software

5.1 Verwendete Engine

5.1.1 Stockfish

Unsere verwendete Engine nennt sich [5]Stockfish Engine. Diese kann auch im [6]Quellcode eingsehen werden, da diese Open Source, sowie frei nutzbar ist. [5]Stockfish braucht sich auch gar nicht hinter kommerziellen Alternativen wie Rybka 4 and Houdini verstecken. [5]Stockfish ist derzeit noch unerreicht was Schachengines im freien Markt anbelangt. Über [8]Twitter gibt es auch von Zeit zu Zeit neue Updates bezüglich der Engine.

Lizenz

Stockfish steht unter der [7]GPLv3 license. Das bedeutet, dass Änderungen nicht nur vorgenommen werden sondern auch frei wieder veröffentlicht werden können. Dies bietet eine gute Grundlage für Weiterentwicklungen. Dies war unter anderem ein Grund zur Wahl dieser Engine.

Tests

Nach ausführlichen Tests der [5]Stockfish Engine mit herkömmlichen angesagten Programmen wie [9]Arena 3.0, kamen wir zum Schluss, dass wir es wirklich mit einer ausgereiften Engine zu tun haben.

Cross Platforms

Ein anderer Grund [5]Stockfish zu wählen war die nahezuhe Unabhängigkeit, gegeben durch die verschiedenen Binaries, welche [5]Stockfish von Hause aus mitliefert. So ziemlich jede große Plattform wird unterstützt. Darüber hinaus gibt es sogar eine [10]App für iPhone, iPod touch, and iPad welche natürlich auf der Stockfish Engine beruht.

Benutzte Version

Unsere genutzte [5]Stockfish Version innerhalb von Twitchess ist die Version [11]2-1-1. Die derzeit neueste Version von Stockfish ist die Verison [12]2-2-1. Ganz nach dem Prinzip never change a running System haben wir daher kein Update vollzogen. Theoretisch ist hier ein Update jedoch ohne Probleme zu vollziehen.

5.2 Verwendete APIs

Wir nutzten folgende APIs

- [13] Twitter4j
- [14] sqlitejdbc
- [15] Universal Chess Interface (UCI)

Beispiel-Kommunikation zwischen der Schachengine und [2]Twitchess (UCIEngine und ucicommands.*)

- > “uci“
- < “[...]“
- < “uciok”
- > “ucinewgame“
- > “isready“
- < “readyok“
- > “position fen rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1“
- > “go movetime 2000“
- < “info [...]“
- < “bestmove e2e4“

Kapitel 6

Anmerkungen

6.1 Kommentare zum Quellcode

Verwendet wird eine Schichtenarchitektur

- Präsentationsschicht: “TwitterView”
- Geschäftslogik: “TwitterBot”, “ChessManager”
- Persistenzschicht: “ChessStateDAO”

[2]Twitchess bekommt eingehende Befehle entweder direkt von Twitter oder von der View und verarbeitet diese. Dazu ruft er den ChessManager auf, der die empfangene Zeichenkette auswertet.

Wir verwenden dazu ein Command-Pattern, der je nach dem ankommendem Befehl die Aufgabe an die entsprechende Klassen

- “CancelGameChessCommand”
- “MoveChessCommand”
- “NewGameChessCommand”
- “OfferDrawChessCommand”
- “PrintGameChessCommand”

deligiert.

Die jeweiligen Befehle greifen über das Interface ChessStateDAOInterface direkt auf die Datenbank zu und holen sich das jeweils aktuelle Spiel.

Der größte Teil der Arbeit wird in MoveChessCommand verrichtet: Dieser greift nicht nur auf die Datenbank zu, sondern muss auch den eingehenden Zug parsen, auf Gültigkeit/Richtigkeit überprüfen, dann die Schachengine befragen und schließlich das Ergebnis persistent in einer SQLite-Datenbank speichern und einen String als Antwort generieren. Die Speicherung der aktuellen Stellung erfolgt als [16]FEN. Dieser wird bei der

Überprüfung des Zuges zunächst einmal in ein Objekt der Klasse “GameState” umgewandelt. Mit Hilfe der Klasse “ChessLogic” und deren Methode “isValidMove(GameState s, Move m)” kann dann getestet werden, ob der auszuführende Zug überhaupt möglich ist. Dabei ist anzumerken, dass wir uns bemüht haben, wirklich jeder Schachregel gerecht zu werden, d.h. wir unterstützen Rochade, Enpassant-Regel und auch das Umwandeln von Bauern nach Erreichen der ersten bzw. achten Reihe in Springer, Läufer, Turm oder Dame.

Literaturverzeichnis

[1] Twitchess Jenkins, http://jenkins.rambow.it:8080/job/Twitchess

[2] Twitchess Github, https://github.com/Mateful/Twitchess

[3] Twitchess Twitter, https://twitter.com/Twitchess1

[4] JAVA Download, https://www.java.com/de/download/

[5] Stockfish Webseite, http://www.stockfishchess.com/

[6] Stockfish Github, https://github.com/mcostalba/Stockfish

[7] GPLv3 license, https://www.gnu.org/copyleft/gpl.html

[8] Stockfishchess Twitter, https://twitter.com/stockfishchess

[9] Arena 3.0, http://www.playwitharena.com/

[10] Stockfishchess App, http://itunes.apple.com/us/app/stockfish-chess/id305558605

[11] Verwendete Stockfishchess Engine , http://blog.stockfishchess.com/post/5316251554/stockfish-2-1-1

[12] Aktuellste Stockfishchess Engine , http://blog.stockfishchess.com/post/15405778824/stockfish-2-2-1

[13] Twitter4j, http://twitter4j.org/

[14] Slitejdbc, http://www.zentus.com/sqlitejdbc/

[15] Universal Chess Interface, http://wbec-ridderkerk.nl/html/UCIProtocol.html

[16] FEN, http://de.wikipedia.org/wiki/Forsyth-Edwards-Notation

Abbildungsverzeichnis