JenkinsBlame

FH Brandenburg - University of Applied Sciences

Wintersemester 2011/2012

Lehrveranstaltung: **Software-Qualität**

Inhalt:

1. Ziel des Projekts
2. Technische Umsetzung
3. Überlegungen
4. How to build
5. Entwickler

1.) Ziel des Projekts

JenkinsBlame hat das Ziel, Entwickler bei Ihrer Arbeit zu motivieren.
Der Service greift dazu per Web-Interface auf einen Jenkins CI Server zu und wertet die einzelnen Build-Prozesse eines Projektes aus.
Schlägt ein Build fehl, so zeigt JenkinsBlame den Verantwortlichen sehr auffällig auf dem Web-Interface an. Ist der Build wieder lauffähig, so wird der Entwickler angezeigt, der das Problem behoben hat.
Zusätzlich werden alle Mitwirkenden eines Projekts ermittelt, und ausgewertet, wer wieviele Builds zerstörte oder fixte. Visualisiert wird dieses Verhältnis mittels eines blauen und eines roten Balkens.

2.) Technische Umsetzung

JenkinsBlame ist ein Service der auf der Google App Engine läuft und ist in der Sprache Java geschrieben.
Zur Speicherung der erforderlichen Daten wird der google-eigene Datenspeicher verwendet.
Dieser unterscheidet sich von gewöhnlichen RDMS, da Google sog. Bigtables einsetzt um einen sehr hohen Grad an Verteilung und Skalierung zu erreichen. Um auf den Datastore zuzugreifen wurde die JDO-API der DataNucleus Access Platform verwandt. Mit einem PersistenceManager werden aus POJOs Entitäten für den Datastore und umgekehrt.

Da im Web-Umfeld üblich und bewährt, wurde das MVC-Pattern vollständig umgesetzt.
Die Model-Komponente wurde in der Klasse JenkinsBlameStats verwirklicht. Sämtliche Datenzugriffe (Google-Datastore, Jenkins CI-Server) erfolgen durch diese Klasse.
Die View-Komponente stellt jenkinsblame.jsp dar und dient lediglich der Interaktion mit dem Nutzer.
Die Controller-Komponente ist das Servlet JenkinsBlameServlet. View und Model kommunizieren ausschließlich über den Controller.

3.) Überlegungen

Google App Engine wurde gewählt, weil damit ein Webserver zur Verfügung steht, der bei geringer Nutzung und wenigen Datastore-Zugriffen, kostenlos ist. Als Testframework sollte Junit eingesetzt werden, wodurch als Programmiersprache Java resultierte. Für die Appentwicklung stellt Google ein SDK und ein Plugin für die eclipse IDE bereit.
Das Jenkins-API liefert entweder XML oder JSON. Schnell wurde deutlich, dass JSON sehr komfortabel und schnell geparst werden kann. Deshalb wurde auf die Verarbeitung von XML verzichtet. Alle benötigten Daten ließen sich mit JSON-Objekten gewinnen.

4.) How to build

JenkinsBlame ist unter folgender URL zu erreichen:

http://jenkinsblame.appspot.com

Es ist nicht nötig JenkinsBlame zu installieren oder zu konfigurieren.
Falls es jedoch gewünscht wird, kann ein Entwicklungsserver gestartet werden oder eine Junit-Testsuit ausgeführt werden. Dazu sind folgende Schritte nötig:

- Clonen des Repository von http://github.com:
  git clone https://github.com/schmidde/JenkinsBlame.git
- Ausführen eines Buildscripts mit Ant:
  ant test (führt Unit- und Integrationstests aus)
  ant runserver (beinhaltet test und startet einen Server auf localhos:8080)

5.) Entwickler:

Dennis Schmidt,
Sebastian Gräbitz