



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektarbeit

**Andreas Müller, Claus Torben Haug, Jan-Dennis Bartels, Marjan
Bachtiari**

MBC-Ping-Pong

Andreas Müller, Claus Torben Haug, Jan-Dennis Bartels, Marjan
Bachtiari

MBC-Ping-Pong

Projektarbeit eingereicht im Rahmen der Wahlpflichtfach

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Martin Becke

Eingereicht am: 27. November 2016

Andreas Müller, Claus Torben Haug, Jan-Dennis Bartels, Marjan Bachtari

Thema der Arbeit

MBC-Ping-Pong

Stichworte

Ping-Pong, NodeJS, JavaScript, WebRTC

Kurzzusammenfassung

In diesem Dokument wird das Projekt im MBC-Ping-Pong, das im Rahmen des Wahlpflichtfaches Modernebrowserkommunikation an der HAW-Hamburg erstellt wird, behandelt. Hierbei handelt es sich um eine Pong Clone welcher auf einem zentralen Bildschirm spielbar ist und mittels WebRTC gesteuert wird. Es wird auf die Architektur, JavaScript, Frontend und Backend eingegangen.

Andreas Müller, Claus Torben Haug, Jan-Dennis Bartels, Marjan Bachtari

Title of the paper

MBC-Ping-Pong

Keywords

Ping-Pong, NodeJS, JavaScript, WebRTC

Abstract

This document is about the Project MBC-Ping-Pong, which is made for the elective course Modernebrowserkommunikation at HAW-Hamburg. Its about a Pong clone, played on a central screen nd controled via WebRTC. The architecture, JavaScript, frontend and backend are discussed.

Inhaltsverzeichnis

Inhaltsverzeichnis	iv
1 Architektur	1
1.1 Arbeitsablauf zur Bearbeitung eines Issue	1
2 JavaScript	4
3 Backend	5
4 Frontend	6
Literaturverzeichnis	8

Listings

1 Architektur

1.1 Arbeitsablauf zur Bearbeitung eines Issue

Um eine erfolgreiche Zusammenarbeit zu gewährleisten, sind allgemein gültige Regeln nötig. Insbesondere wird festgelegt, wie die einzelnen Arbeitsschritte ablaufen sollten, um ein Issue zu bearbeiten. Zudem werden weiterhin die Zuständigkeiten geregelt.

Verwaltung der zu bearbeitenden Issues

Die zu bearbeitenden Issues werden auf GitHub unter Issues (<https://github.com/Transport-Protocol/MBC-Ping-Pong/issues>) gepflegt. Um den Verlauf eines Issues darzustellen wird das Kanbanboard von GitHub (<https://github.com/Transport-Protocol/MBC-Ping-Pong/projects>) genutzt.

Erstellen der zu bearbeitenden Issues

Prinzipiell kann und darf jedes Projektmitglied zu jeder Zeit Issues erstellen. Gerade bei Bugs ist dies ein gewünschtes vorgehen. In der Regel sollten dies jedoch aus Gruppensitzungen hervorgehen und durch den Architekten ausformuliert werden.

Ein Issue besteht aus drei Absätzen:

- **Beschreibung**

In der Beschreibung wird allgemein auf den Kontext des Issues eingegangen.

- **Anforderung**

In Anforderung wird die Zielvision dargestellt.

- **Abnahmekriterien**

In Abnahmekriterien werden alle Punkte aufgeführt, die notwendig sind, um das Issue als erfolgreich bearbeitet anzusehen.

Das Kanbanboard

Das Kanbanboard ist in fünf Abschnitte eingeteilt:

- **Selected for Development**

Diese Spalte enthält alle Issues, die der Architekt zur Bearbeitung in nächster Zeit ausgewählt hat. Hier enthaltene Issues sind entweder durch den Architekten einem bestimmten Teammitglied zugeordnet. Diese sollten dann auch vorrangig bearbeitet werden. Oder (dies sollte der Normalfall sein) sie sind niemandem zugeordnet, dann kann sich jedes Teammitglied entscheiden, ob er das Issue bearbeitet. Gründe für das direkte zuweisen können unter anderem sein, dass es eine entsprechende vorhergehende Absprache gab, dass der Architekt das Issue speziell einem Bereich zugehörig sieht bzw. eine spezielle Paarung erreichen möchte, oder aber auch, weil ein Issue schon zu lange in "Selected for Development" verweilt. Hat sich ein Teammitglied für ein Issue entschieden, trägt er sich als Bearbeiter ein und zieht es in auf "In Development".

- **In Development**

In dieser Spalte verweilen alle Issues, an denen gerade entwickelt wird. Wenn die Entwicklung an einem Issue abgeschlossen ist, zieht der Bearbeiter das Issue weiter auf "Needs Review".

- **Needs Review**

Hier verweilen alle Issues, deren Entwicklung abgeschlossen ist, aber noch nicht geprüft wurde, ob die Abnahmebedingungen erfüllt sind. Normalerweise sollte die Abnahme durch den Architekten erfolgen. Issues, die der Architekt bearbeitet hat, muss das Review von einem anderen Teammitglied gemacht werden. Ein Issue bei dem das Review durchgeführt wird, wird in die Spalte "In Review" verschoben.

- **In Review**

Hier sind alle Issues enthalten, die sich gerade im Review befinden. Sind alle Abnahmekriterien erfüllt, und sind durch die Bearbeitung des Issue keine neuen Probleme/Fehler hinzugekommen, wird es in die Spalte "Done" verschoben und das Issue geschlossen. Ist dies nicht der Fall, wird ein Entsprechender Kommentar mit einer möglichst detaillierten Beschreibung des Problems an das Issue angehängt, und es wieder auf "In Development" geschoben.

- **Done**

Diese Spalte enthält alle abgeschlossenen Issues.

Git

Hier sind die Verhaltensweisen für die Nutzung von Git aufgeführt. Alles hier nicht aufgeführte kann von jedem Teammitglied nach eigenem Ermessen gehandhabt werden.

- **Branches**

Für jedes Issue wird ein Branch erstellt, außer es handelt sich um reine Dokumentation (im Ordner Docu). Ein Branchname folgt folgendem Muster: "#<IssueNummer> <KurzerName>". Dadurch lässt sich

- **Commits**

Commits folgen folgendem Namensschema: "#<IssueNummer> <Beschreibung>".

- **Push und Pull**

Es sollte möglichst häufig gepusht werden, um einen eventuellen Datenverlust zu vermeiden. Beim Pull sollte mit -rebase gearbeitet werden, um die Historie möglichst sauber zu halten.

- **Merge und Pullrequest**

Bevor ein Issue auf "Needs Review" geschoben wird, ist der Master in den Branch zu mergen und ein Pullrequest (<https://github.com/Transport-Protocol/MBC-Ping-Pong/pulls>) zu erstellen. Derjenige, der das Issue reviewt hat, merget den Branch dann mithilfe des Pullrequests in den Master und löscht ihn.

2 JavaScript

3 Backend

4 Frontend

See also [One und Two \(2010\)](#).

Literaturverzeichnis

[One und Two 2010] ONE, Author ; TWO, Author: A Sample Publication. (2010)

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 27. November 2016

 Andreas Müller, Claus Torben Haug, Jan-Dennis Bartels, Marjan Bachtiri