Reich werden in 3 einfachen Schritten

MVC Pattern und Events in Python & Javascript

Der Plan

- Schicken online Chat bauen
- ???
- Reich werden!

Die Anforderungen

- Layout und Code sollen möglichst getrennt sein
- Der Chat muss geringe Latenz haben
- Der bahnbrechende Erfolg darf nicht durch mangelnde Skalierung gefährdet werden

Die Probleme

- jQuery und Javascript laden ein zum Spaghetti-Coden
- HTTP kennt keine Push-Events
- WSGI Prozesse skalieren nicht

Spaghetti mit jQuery-Sauce

- Simpler Javascript/jQuery-Code läd dazu ein, Daten und DOM zu vermischen
- $(<selector>).data(“foo”, bar)
- Bei Änderungen des Layout leidet die Logik, und umgekehrt

Rückgrat zeigen

- backbone.js separiert Daten von Präsentation
- Unterstützt Modelle und Collections
- Schickt Events bei Änderungen
- Views reagieren auf die Events

Zeit für Code

- User-Listen

Push Events

- WebSockets: schöne, neue Welt
- Leider doch nicht
- Doch es gibt eine Lösung

Der Lange Marsch

- Long Poll/Comet ist die Lösung
- Zwei Implementation: XHR und Script-Tag
- Ich verwende XHR

Zeit für Code

- Die Client-Seite

Stürmisch skalieren

- Long Poll braucht hält viele Verbindungen lange auf
- Blockierende Technologien wie Apache & WSGI sind dafür zu Resourcen-intensiv
- Event-basierte Server sind die Lösung

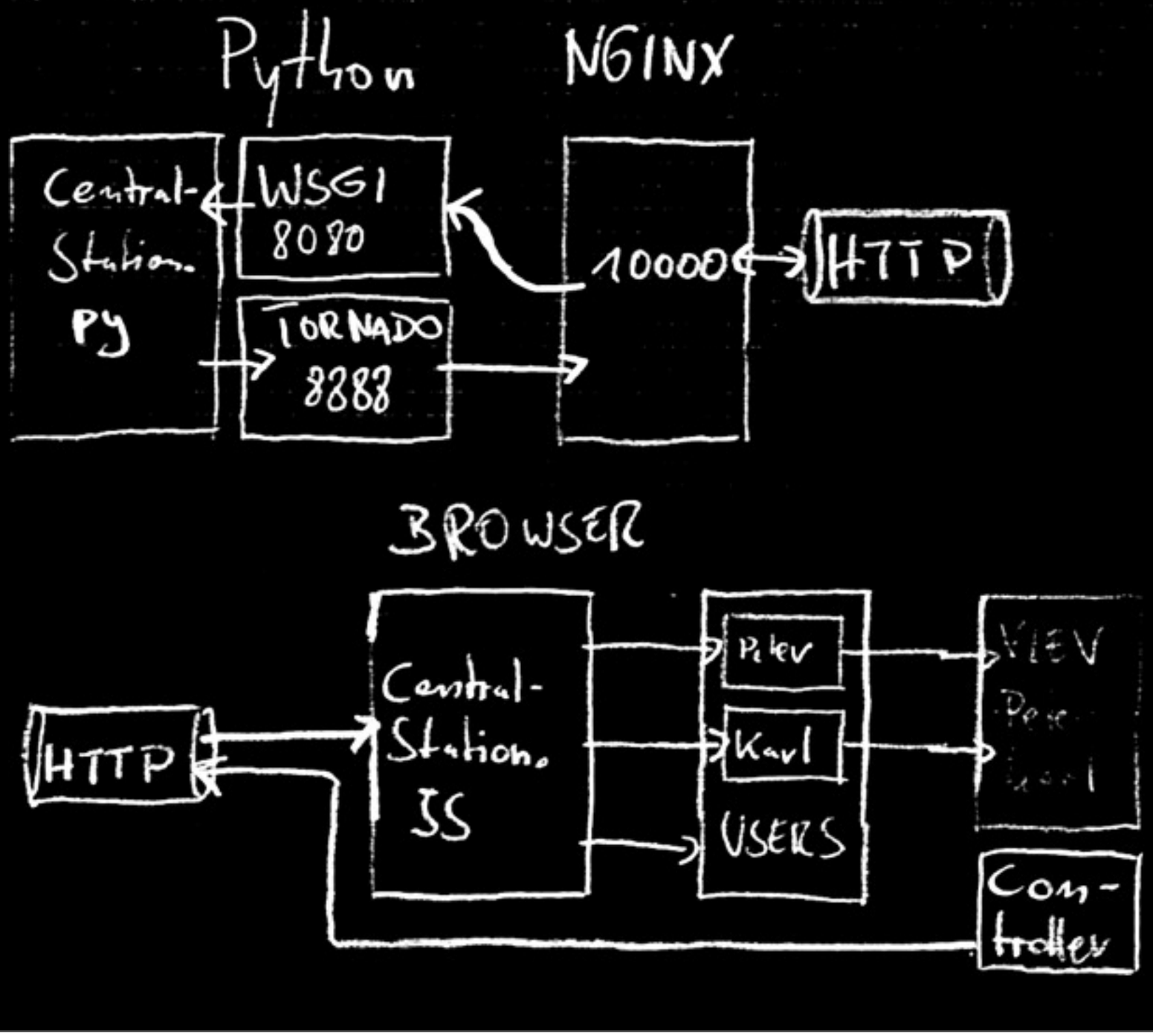
Tornado

- Ursprünglich von FriendFeed.
- Asynchron/Event-Basiert

Zeit für Code

- Hin
- Und zurück

Der Überblick



Ausblick

- Serverseitige Modelle & Collections
- Asynchrones backbone.js
- Verteilter Zustand serverseitig

Letzte Worte

- Wer selbst spielen will: hier