Seminar TIE-11406: Backend as a Service (BaaS)

Tampere University of Technology
01.10.2013

http://www.deployd.com/

technology presentation

Juha Nurmi, Timo Kokko

Deployd - technology presentation

1. Introduction
2. Overview of the system and its design principles
3. Existing demos
4. Walkthrough of a complete example
5. Evaluation (benefits, drawbacks, usefulness, possible measurements)
6. Summary

Introduction

Deployd - what is it?

- Backend as a service
- homepage: http://www.deployd.com/
- An open source platform
- Apache license version 2.0
- GIT: https://github.com/deployd/deployd
- Host it yourself ~~or use Deployd's cloud~~
- Documentation ~~& community~~

Deployd - what is it?

- Deployd is designed to replace traditional backends for web and mobile apps
- Perform advanced queries over HTTP
- Push data to clients over WebSockets in real time
- Transform and validate your data **as it changes**

Why Deployd?

Build apps, not backends.

- By removing the complexity of backend development, Deployd lets you focus on the part of your app that really matters: the front-end.

Deploy your app effortlessly.

- Just click a button and the latest version of your app is online. Show your friends what you've made without the pain of setting up servers and databases.

Team Tailored Experience

- Collaboration and version control
  - Everything on the dashboard is separated into JSON and JavaScript files in the filesystem which can be easily version controlled
- Develop apps locally
  - Develop apps locally and deploy when ready
- Open source
  - Apache 2 license
  - Modify and run as you like
- Static file support
  - Host static HTML, JavaScript, images, and other static assets under /public directory

The Simplest Way To Build API

1. No Boilerplate
    - API up and running with one command
    - Empty canvas waiting to add resources
2. Resources
    - Resources can be defined through Deployd dashboard
3. Dashboard
    - Add and manage APIs through web- based dashboard
4. 1-Step Deploy
    - Deploy API to scalable cloud or host it yourself

Do More With Less

- One API for both server side and client side code.
- Store your data as JSON objects.
- Run advanced queries directly from your mobile or JavaScript app.
- Create and authenticate users without any setup or boilerplate code.
- Easily validate and transform data as it changes.
- Keep all your clients in sync in realtime with a single line of JavaScript.
- Serve your app's files without any setup.
- Insert, Update and Delete Objects from your client without a custom web app or api.
- Extend Deployd by installing modules that expose useful apis to your clients

Flexibility Through Modularity

- Consists of core library with modular API to extend application
  - Quickly add custom client-facing Resources
  - Integrate third-party services or APIs
  - 17 000 + node modules to extend API
- Resource distribution feature
  - Makes easy to add external modules to app
  - Currently under development

Death to Polling

- Keep all client application in sync
- Avoid needless refreshes with Deployd's real-time capabilities
- Easy to listen and respond to changes in app using dpd.js or another client which can support websockets
- Easy to validate, secure, and scope real-time messages to specific users or groups

Activity in Github

- 1122 commits
- 13 branches
- 38 releases
- 16 contributors

Overview of the system and its design principles

Javascript Throughout

- Build on Node.js and MongoDB
- Server-side build with JavaScript
- Unified dpd.js client/server library
- Real-time capability through Websockets
- All data stored as JSON
- API endpoints also exposed over REST

Other Features

- Unified API to app resources on client and server
  - New resource added to API is automatically available via dpd ['resourceName']
- Flexible validation
  - Javascript code inside of events to validate users role and request
- Easy graph data
  - Multi-dimensional data through relating and embedding data
- Out of the box user management
  - Custom properties and roles to users
  - Custom event scripts to control access to users
- Client-side advanced queries
  - Perform queries against Collections in the client

Getting started with Deployd

- Host it yourself ~~or use Deployd's cloud~~
- Get the source from the GIT
- Basically install it to your Linux server
- Build on MongoDB 2.0.x and NodeJS 0.8.x

Getting started with Deployd

- Web-based IDE (Dashboard)
- You are working with JSON+JavaScript
- Or just using JSON HTTP API without JavaScript

Existing demos

Existing demos

- Many basic demos, for instance,
  - todo list demo
  - login-form demo
  - chatroom demo

Check what the browser does

- Data is stored dynamically in the JavaScript

→if the sending to the backend fails the data is lost when the browser is closed

- It is send to the backend using HTTP POST

→ webSockets can be used to listening events

Walkthrough of a complete example

Walkthrough of a complete example

- See Deployd's documentation
- # dpd create comments
- # cd comments
- HTML and JavaScript files to /public/
- # dpd --open --port 80
- See the result http://37.252.125.242:2403/
- Create a new collection using the dashboard http://37.252.125.242:2403/dashboard/
- /resources/comments# cat config.json

Evaluation

Evaluation

- benefits: open source, manage your data
- drawbacks: command line does not work :(
- usefulness: easy to use

Does it scale?

- MongoDB and Node.js are scalable

  → Unfortunately, there is no documentation how to scale Deployd application

Evaluation: performance

- We know that Node.js and MongoDB have good performance
- Let’s test the full Deployd system
- Using comments demo
- How the client side and server side performs
- Add 1000 lines of data → small load
- Add 10 000 lines of data → disaster

Evaluation: performance

```
var MAX = 10000; //destroys the whole Deployd system, reboot does not help!
var MAX = 1000; //handles without problems and without consuming CPU/RAM
for( var i = 0; i<MAX; ++i) {
  var name = Math.floor(Math.random()*1000001)+"x";
  var comment = Math.floor(Math.random()*100000001)+"x"
  dpd.comments.post({
        name: name,
        comment: comment
  }, function(comment, error) {
        if (error) {
         console.log( error );
        }
  });
}
```

Evaluation: performance

(node) warning: Recursive process.nextTick detected. This will break in the next version of node. Please use setImmediate for recursive deferral.

RangeError: Maximum call stack size exceeded

→ **WTF? Kills the whole Deployd permanently**

Summary

Summary

- Open source
- You can host it yourself and control your data
- Good documentation
- Easy to use
- Problems with heavy load :(
- Command line does not work :(
- How to scale?
- Future? Community? Development?

Possible technology demonstration

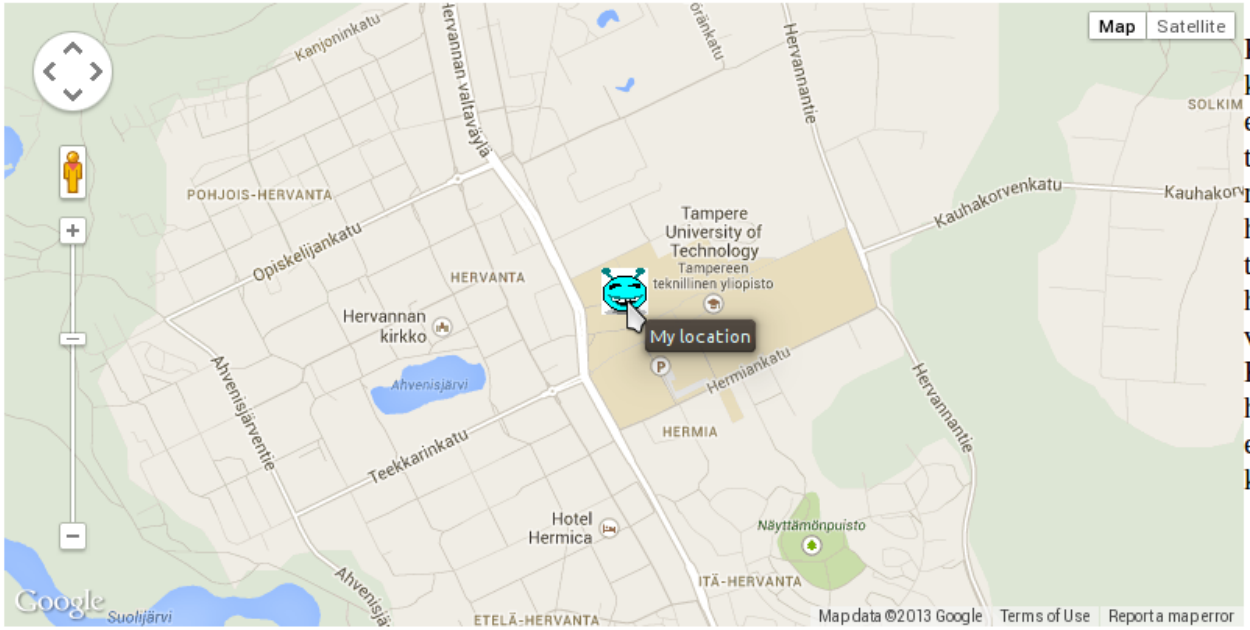
Motivation

- http://apps4finland.fi/
- https://github.com/apps4finland/haaste-kaupungin-viat
- http://dev.hel.fi/apis/issuereporting

→ why not to create a map based issue tracking system?

Issue tracking system for Tampere

Kunnossapitoa ja korjaamista kommunikoiden



Kaupunkiympäristössä on jatkuvasti jotain korjattavaa. Huolellakin suunnitellut asiat eivät aina toimi niin kuin alunperin on tarkoitettu kun ympäristöolosuhteet muuttuvat. Kaupungilla liikkuvat ihmiset huomaavat monasti puutteet, viat ja toimattomuudet nopeasti, ja voisivat havainnoillaan tehostaa ylläpidosta vastuullisten organisaatioiden toimintaa. Kaupunkilaiset voivat vaivattomasti jakaa havaintonsa ja tietonsa korjattavista epäkohdista kaupungin organisaatioiden kanssa tämän kartan avulla.

Raahaa sininen otus siihen kohtaan karttaa, johon tekemäsi ilmoitus liittyy.

Nimimerkki: Nimimerkki

Ilmoitus:

Kommentti

Lähetä

Issue tracking system for Tampere

- Will be BSD licenced and free
- Any city can take and use it :-)
- Client side
  - JavaScript + HTML5
  - HTML5 geolocation
  - Google Maps JavaScript API v3
- Server side
  - Deployd
    - Chat nick name, location, messages...
    - Integration to existing open source ticket system