

# The Gears of Minkepedia.org

An introduction to the  
technologies and design principles  
behind Minkepedia

Hans-Joachim Belz  
[achim@minuteefforts.com](mailto:achim@minuteefforts.com)

# Agenda

This is a brief overview of the technologies and basic design principles behind the social knowledge mapping platform Minkepedia.

For an introduction to what Minkepedia actually wants to achieve for its users, see theses slides:

<http://www.minkepedia.org/slides/minkepedia-slides.pdf>

Or visit <http://www.minkepedia.org>

# Main Drivers: User Experience

- Rich visual user experience
  - Explorative Prototyping with RAD tool
  - Interactive knowledge maps make up the core of the UI
- Aggressive use of available web technologies  
=> support only cutting edge browsers
- Support for mobile use of the platform  
(both via web and native apps)



No IE 6!  
Hell, no IE 8,  
either!

# Main Drivers: Minimal Effort

- Minimal investment into infrastructure
  - Platform Administration  
(setup, update, backup, security, etc.)
  - Standard Functionality  
(e.g. account management, messaging, etc.)
- Smooth ramp up of load (user and data)
- Cheap scalability for peak times
- Design cheating for minimum working efforts
  - Take deliberate shortcuts to working functionality
  - Fix bad smells when the new functionality has proven worth the effort



Popularity  
always comes  
in hiccups.

# Technology Decisions

PaaS



Google App Engine

Rich  
Client  
Platform



Google Web Toolkit

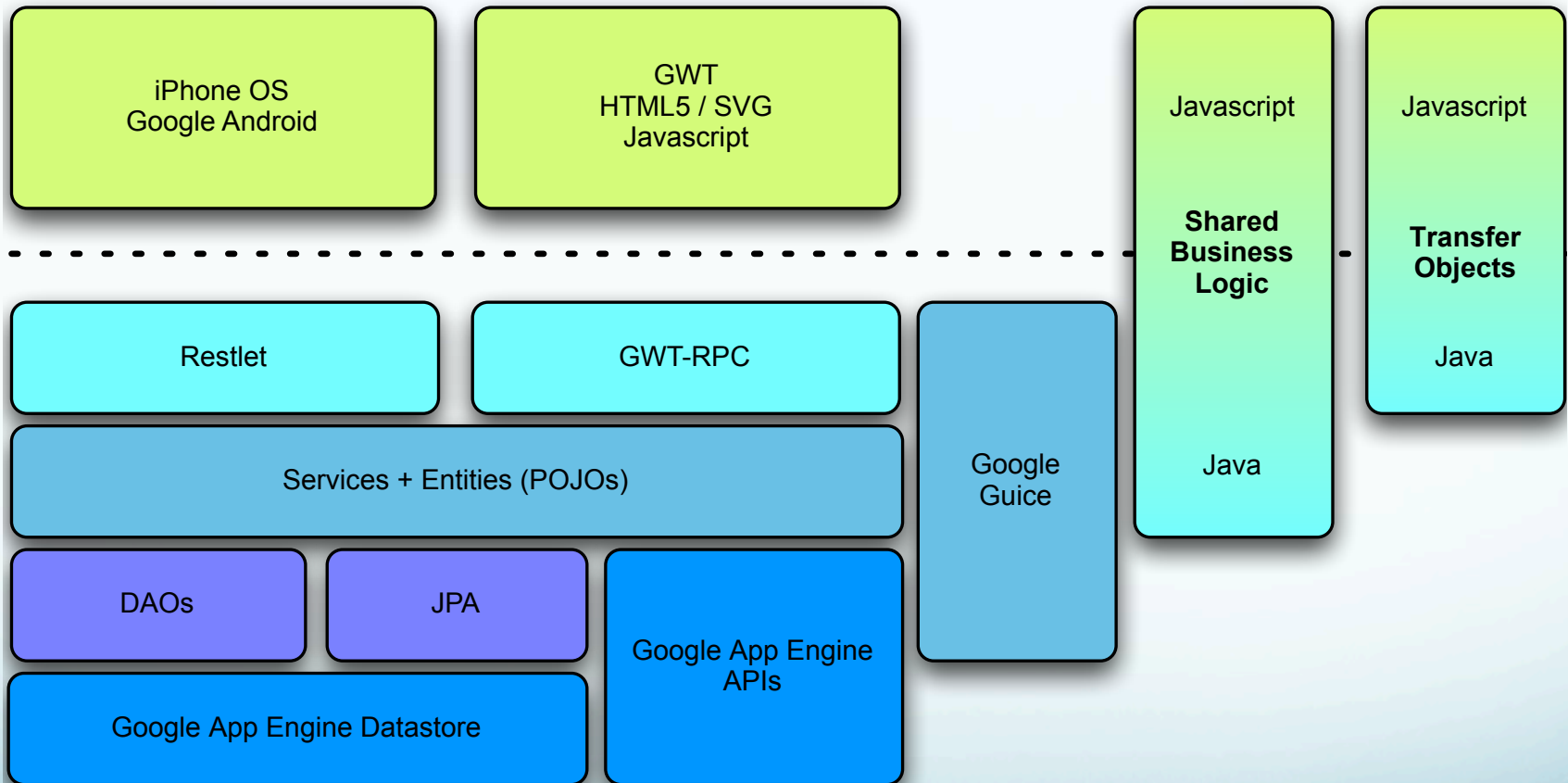
RESTful  
Web  
Services

**Restlet**

Interactive  
Visualization

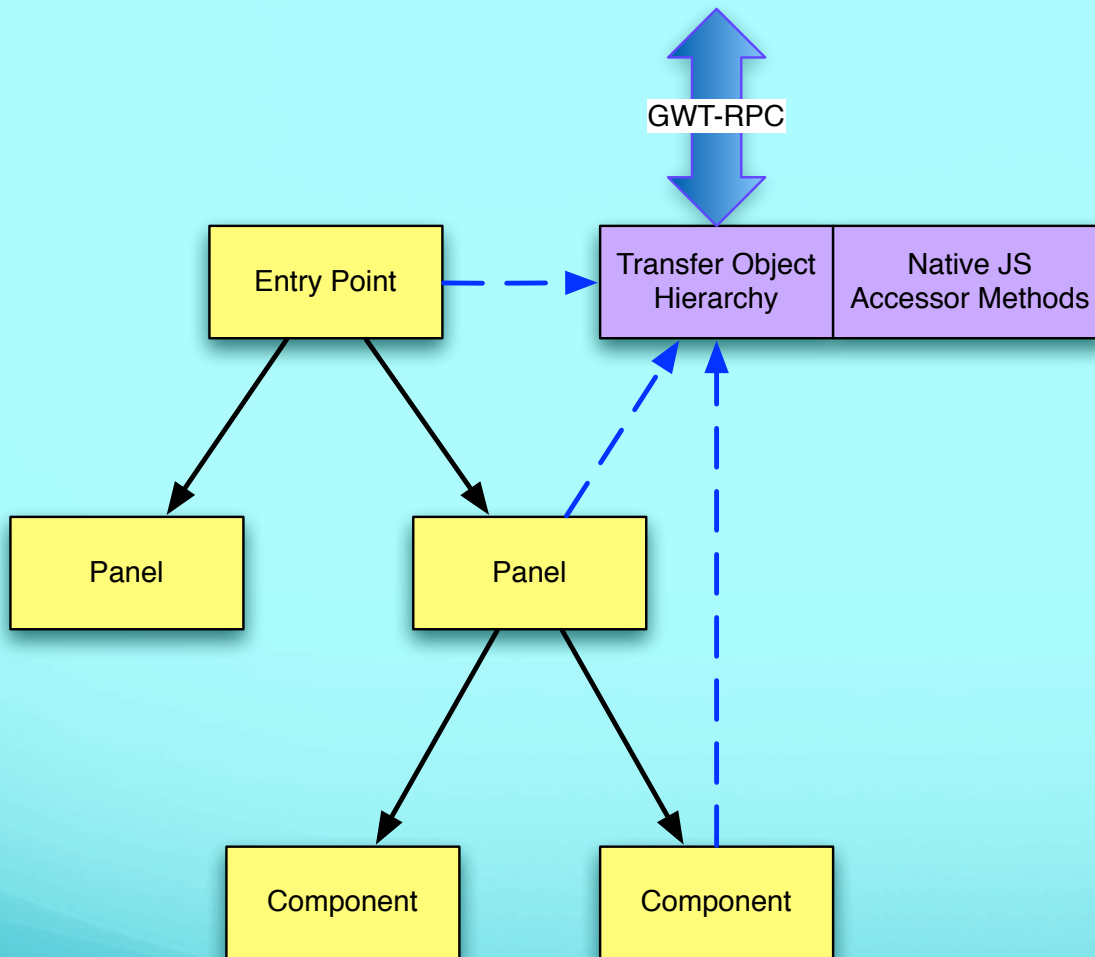


# Technology Stack

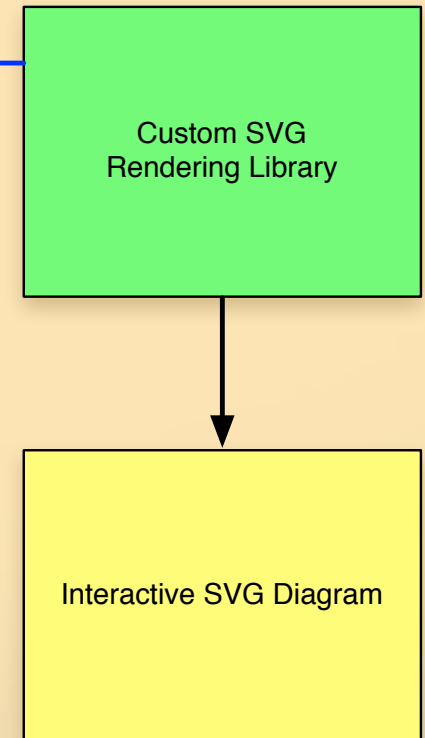


# The web frontend

## HTML + GWT (Java compiled to Javascript)

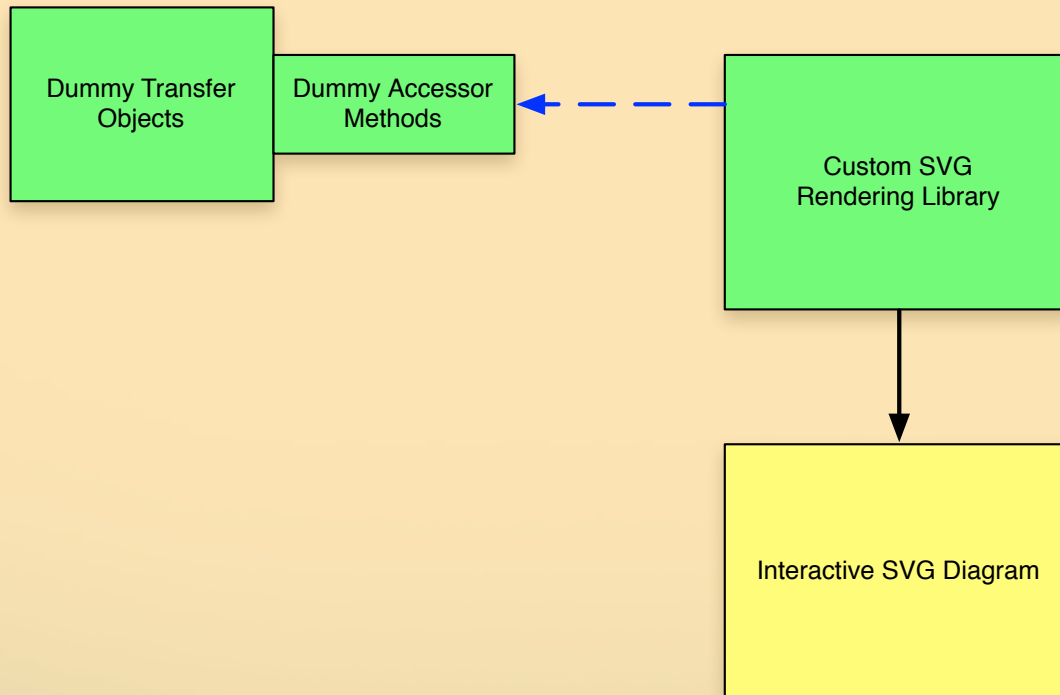


## SVG + native Javascript (iframe)



# Mocked SVG Development

SVG + native Javascript (iframe)



- Static SVG file
- Mocked map data
- No server needed
- No GWT knowledge needed
- Easy definition of test cases for the rendering lib



# Current Security Model

## Service Layer

Services  
(POJO)

Authorized for writing maps (Guice AOP)

## Frontend Layer

Static Content

(HTML, CSS, SVG,  
JS, images, etc.)

/login & /system

(e.g. error page)

GWT Host Pages

(JSP not HTML!)

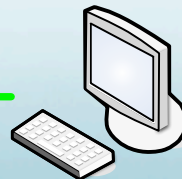
GWT-RPC

REST API

App Engine Cache  
(appconfig.xml)

Registered (ServletFilter)

Authenticated (Google App Engine - web.xml)



# Security with unrestricted read

## Service Layer

Services  
(POJO)

Authorized for read / write (Guice AOP)

## Frontend Layer

Static Content  
(HTML, CSS, SVG,  
JS, images, etc.)

Other JSP  
(e.g. login and  
registration pages)

GWT Host Pages  
(JSP not HTML!)

JS-based hiding of  
functionality  
(no real security!)

GWT-RPC

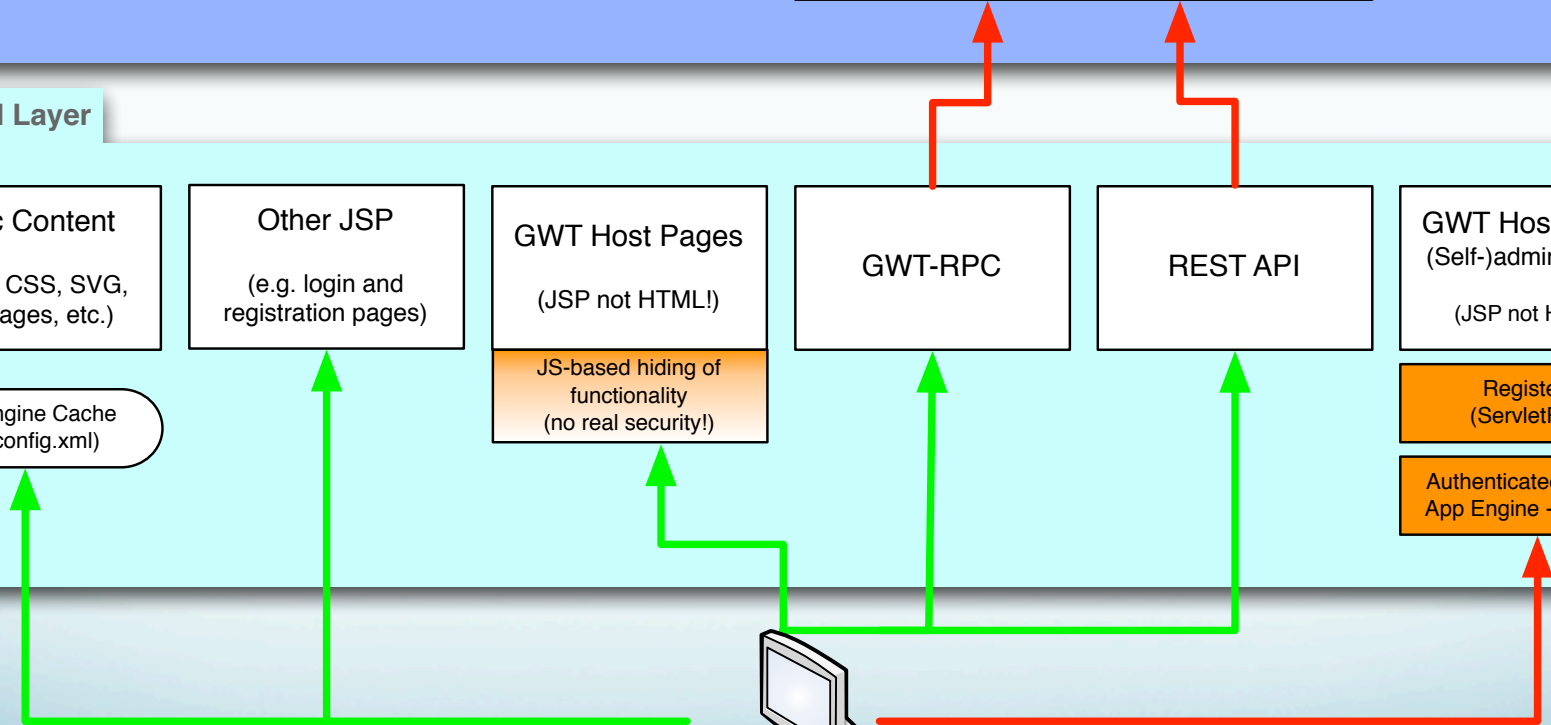
REST API

GWT Host Pages  
(Self-)administration  
(JSP not HTML!)

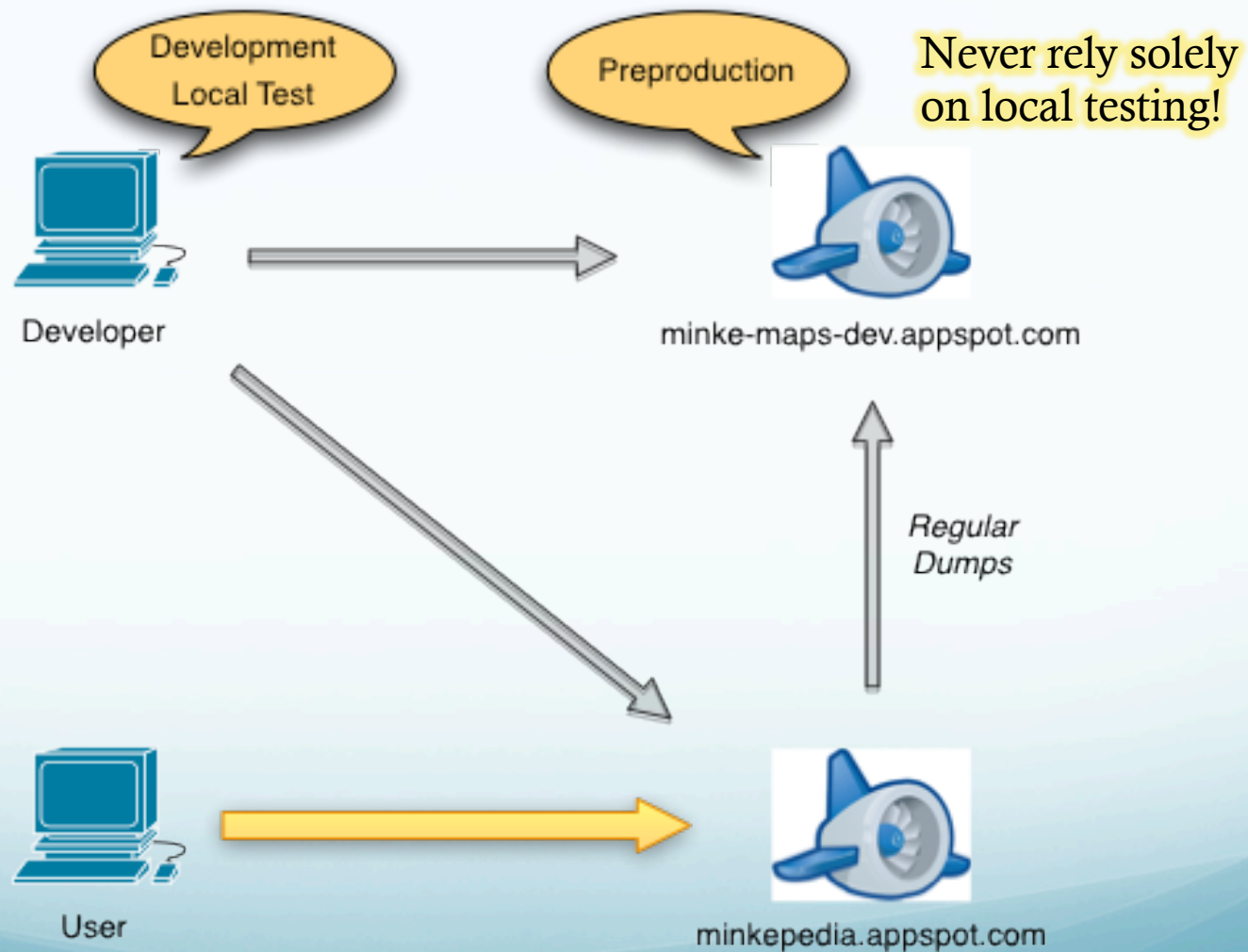
Registered  
(ServletFilter)

Authenticated (Google  
App Engine - web.xml)

App Engine Cache  
(appconfig.xml)



# Test & Deployment



# Roadmap

- Next Steps:
  - General read-only access
  - New UI (support for mobile Webkit-based browsers)
  - Clean up of service layer
  - Maven-based build



➔ Launch of the OpenSource Project (Q2/11?)

**General public access** only after implementing safeguards against vandalism!

# Thank you for your attention!

Stay in touch:

<http://www.minkepedia.org>

<http://www.twitter.com/minkepedia>

[achim@minuteefforts.com](mailto:achim@minuteefforts.com)



The content of these slides is licensed under the Creative Commons Attribution-NoDerivs 3.0 License.  
©minuteFForts (Hans-Joachim Belz) 2011