



QWC2 Viewer für QGIS Server mit Microservice-Architektur

Pirmin Kalberer
@implgeo
Sourcepole AG, Zürich
www.sourcepole.ch





OGIS Web Client 2

- **Ziele OGIS Web client (OWC2):**
 - Modernes, responsive UI
 - Fokus auf Usability
 - Skalierbare Architektur
 - Modularer Code
 - State of the art Technologie: ReactJS, WebPack, OpenLayers 6





OGIS Web Client 2





OGIS Web Client 2

kanton glarus
geodatenviewer

Adressen, Flurnamen, Grundstücke, Koordinaten...

Karten & Werkzeuge

- Kartenthema auswählen
- Kartenebenen zusammenstellen
- Karten-Link erstellen und teilen
- PDF-Karte oder Rasterbild drucken
- Kartenwerkzeuge
- Hilfe, Dokumentation, Impressum

Werkzeuge
Die einzelnen Werkzeuge werden wie bisher als frei schwebende Panels geöffnet oder nach unten aufgeklappt

Hintergrundkarten

- kein Hintergrund
- Luftbild (Orthofoto)
- farbige Karte
- Graustufenkarte

2 km

Koordinaten [m]: 2713353 / 1210363

Geoportal Kanton GL Nutzungsbedingungen



QGIS Web Client QWC2 (minimal)





- › **Kartenkomponenten**
 - › ReactJS + Redux
 - › OpenLayers 6
- › **Build tool chain: nodejs / yarn / webpack**
 - › QWC2 demo Applikation als Basis
- › **QGIS Server**
- › **Optional: Serverseite Komponenten für Suche, Permalink, etc.**



Modulare Kernfunktionen

- › Themenwähler (Projekte)
- › Layerbaum
- › Feature info
- › Suche mit konfigurierbaren Providern
- › Messwerkzeuge
- › Redlining Funktionalität
- › Permalink Generierung
- › PDF-Druck
- › Screenshot / Raster Export
- › WMS / WFS Import
- › KML Import
- › Kartenvergleichs-Tool



Erweiterte Funktionen

- › **Funktionen mit Server-Komponenten**
 - › DB-Suche
 - › Permalinks
 - › Reporting
 - › Editierfunktionen
 - › Mapinfo-Service (layerunabhängig)
 - › Legendendruck
 - › Höhenprofile
 - › Selbstregistrierung (Gruppen)
- › **Übersetzt in 7 Sprachen**



Demo



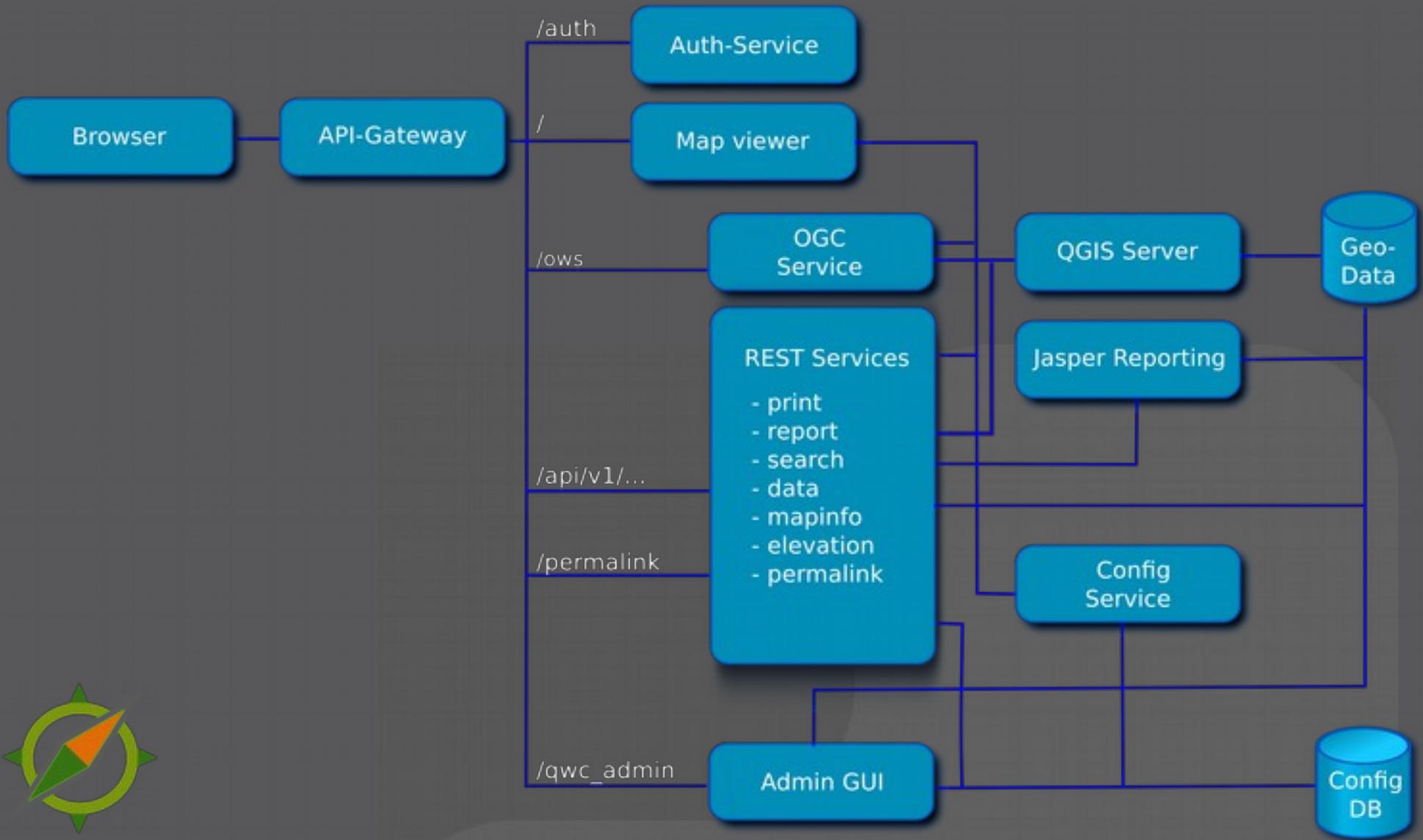
- › **Optionales Backend für QWC2**
- › **Modular, Micro-service orientiert**
- › **(Hauptsächlich) Python Flask Web Applikationen**
- › **Integrierte API Dokumentation (OpenAPI/Swagger)**
- › **Deployment als Docker Container oder WSGI**



The screenshot shows the GitHub repository page for 'qwc-services'. At the top, there is a navigation bar with 'Search or jump to...', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The repository name 'qwc-services' is prominently displayed with its logo, a green compass rose. Below the name, there are tabs for 'Repositories 15', 'People 3', 'Teams 1', 'Projects 0', and 'Settings'. The 'Pinned repositories' section features two cards: 'qwc-services-core' (Python, 1 fork) and 'qwc-docker' (Shell, 1 star, 1 fork). A search bar 'Find a repository...' is present, along with filters for 'Type: All' and 'Language: All', and a 'New' button. The main content area lists 'qwc-config-service' (Python, updated 7 days ago) and 'qwc-map-viewer' (Python, updated 8 days ago). On the right, there are sidebars for 'Top languages' (Python, Shell, Java) and 'People' (3), featuring the profile of 'manisandro' (Sandro Mani).



Server Architektur





Kartenviewer und Drucken

- › **QWC map viewer**
 - › QWC2 viewer service mit Zugriffskontrolle
- › **QWC OGC service**
 - › QGIS Server basierter WMS/WFS mit Zugriffskontrolle
- › **QWC Print service**
 - › QGIS Server basierter PDF Druck mit Zugriffskontrolle



Weitere Dienste

- › **QWC Search Service**
 - › Such-API für benutzerdefinierte Suche
- › **QWC Data Service**
 - › Lesen und Editieren von Daten
- › **QWC Mapinfo service**
 - › Layerunabhängige Info (Rechtsklick)
- › **Permalink service**
- › **Elevation service**



QWC Admin GUI

- › **Web-GUI für Konfigurations-DB**
- › **User und Gruppenmanagement**
- › **Rollenbasierter Zugriff auf Ressourcen**
- › **Ressource-Typen:**
 - › Karten (QGIS-Projekte)
 - › Drucktemplates
 - › Layer, Attribute
 - › Daten (create, read, update, delete)
 - › Viewer, Viewer tasks
 - › Benutzerdefinierte Ressourcen



Authentisierung

- › **Modulare Authentisierungs-Dienste**
 - › Stellen JWT Tokens für zugriffsbeschränkte Zugriffe aus (Cookie oder JWT token header)
- › **QWC DB Auth**
 - › Authentisierung mit User-DB
- › **QWC LDAP Auth**
 - › Authentisierung mit LDAP/Active Directory
- › **und weitere (z.B. Kerberos, SAML)**

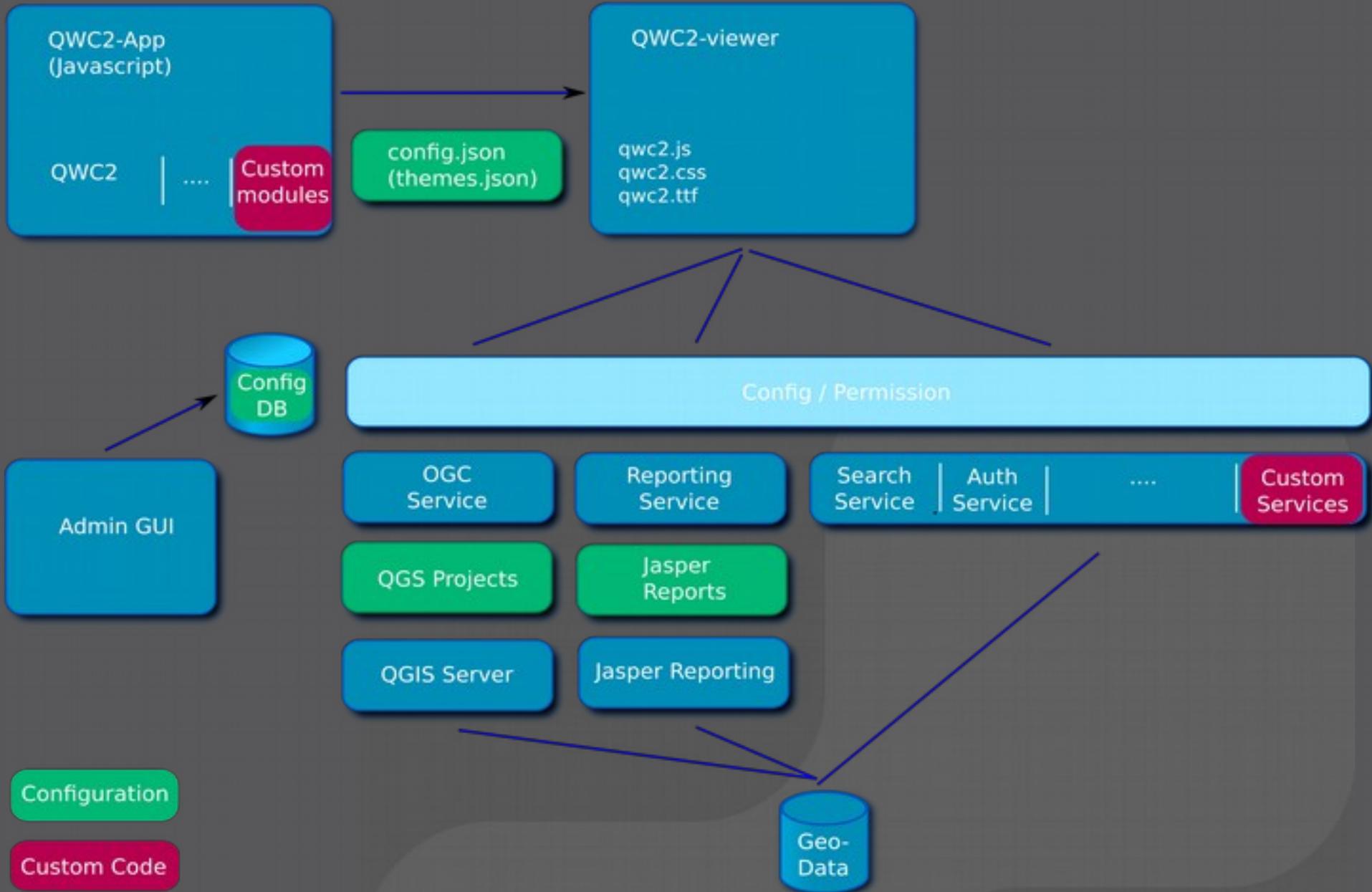


Interne Dienste

- › **QWC Services Core**
 - › Basismodul für QWC services und Dokumentation
- › **QWC Config Service**
 - › Zugriffsrechte und User-spezifische Konfiguration
- › **QGIS Server**
- › **Jasper Reporting Service**
 - › Jasper Reports web service



Customizing





Micro Services

- **Komplexe Anwendung aus lose gekoppelten Teilprozessen (Services)**
- **Isolierte, überschaubare Funktionalität in Services**
- **Kommunikation zwischen Services über HTTP/REST**
- **<https://de.wikipedia.org/wiki/Microservices>**



Docker / Kubernetes

› Docker

- › Bereitstellung von Anwendungen mit Containervirtualisierung
- › 1 Service pro Container
- › Einfache Skriptsprache zur Erstellung
- › Verteilung von Images über Registry
- › <https://www.docker.com/>

› Kubernetes

- › System zur Automatisierung der Bereitstellung, Skalierung und Verwaltung von Container-Anwendungen
- › <https://kubernetes.io/>



Vor-/Nachteile Micro-Services

- 👎 **Aufwändiger Betrieb (Anzahl Services, DB-Connections, Logs, ...)**
- 👍 **Feingranulare Skalierbarkeit**
- 👎 **Separierung Authentisierung+ Autorisierung**
- 👍 **Modulare Authentisierung**
- 👍 **Services mit verschiedenen Technologien**
- 👍 **Modulare Migration von Services**
- ...



› Quellcode und Issue Tracker

- › <https://github.com/qgis/qwc2-demo-app>
- › <https://github.com/qgis/qwc2>
- › <https://github.com/qwc-services/qwc-services-core>
- › <https://github.com/qwc-services/qwc-docker>

› QWC2 Dokumentation

- › https://github.com/qgis/qwc2-demo-app/blob/master/doc/QWC2_Documentation.md

› Beispiele:

- › <https://map.geo.gl.ch>
- › <https://geo.so.ch/map/>
- › <https://qgiscloud.com>



Danke!



Pirmin Kalberer
@implgeo