

DIE KARTE VERÄNDERT SICH DER STANDARDSTIL OPENSTREETMAP- CARTO

Michael Glanznig

a.k.a. nebulon42 ([OSM](#), [GitHub](#))

EIN BISSCHEN KONTEXT

- aktiver Mapper seit 2013
- seit Ende 2014 aktiv bei openstreetmap-carto (Icons...)
- Beiträge zu Kosmtik und Magnacarto
- seit Kurzem Committer bei Mapbox' CartoCSS

keinerlei Entscheidungsbefugnis bei openstreetmap-carto

DIE KARTE VERÄNDERT SICH*

am Beispiel *Salzburg*

* orig. [The map you see on OpenStreetMap.org is changing](#)

ZEITRAFFER AUF VIMEO

- Salzburg Nonntal (16/47.7940/13.0525)
- Salzburg Staatsbrücke (18/47.80136/13.04418)
- Daten vom 30.05.2016
- lizenziert unter CC0



v2.35.0

ZWECK/ZIELE VON OSM-CARTO *

- primärer Feedback-Mechanismus für Mapper (Detail, Kontrast)
- primäre Visualisierung für Besucher auf osm.org (Design, Ästhetik)

Diese Ziele widersprechen sich teilweise.

siehe [Cartography guidelines](#)

**SPANNUNGSFELD
FEEDBACK-MECHANISMUS
VS.
VISUELLE ÄSTHETIK**

WAS BEDEUTET FEEDBACK-MECHANISMUS?

Mapper verwenden osm-carto, um ihre Arbeit zu verifizieren.
Wird es dargestellt? Wird es *richtig* dargestellt?

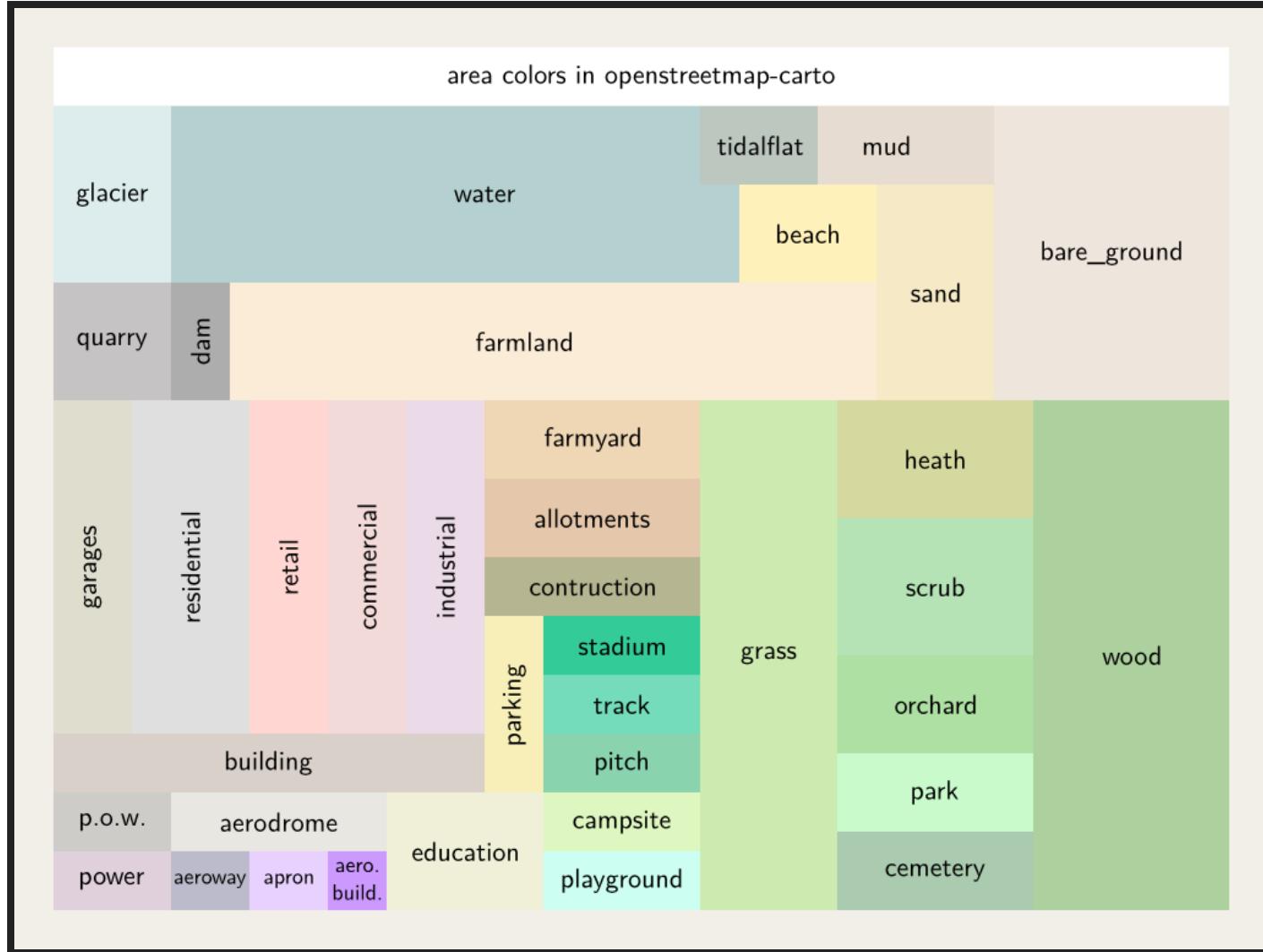
opt. Wenn es nicht dargestellt wird, was kann ich machen, damit es dargestellt wird?

⚠ Taggen für den Renderer

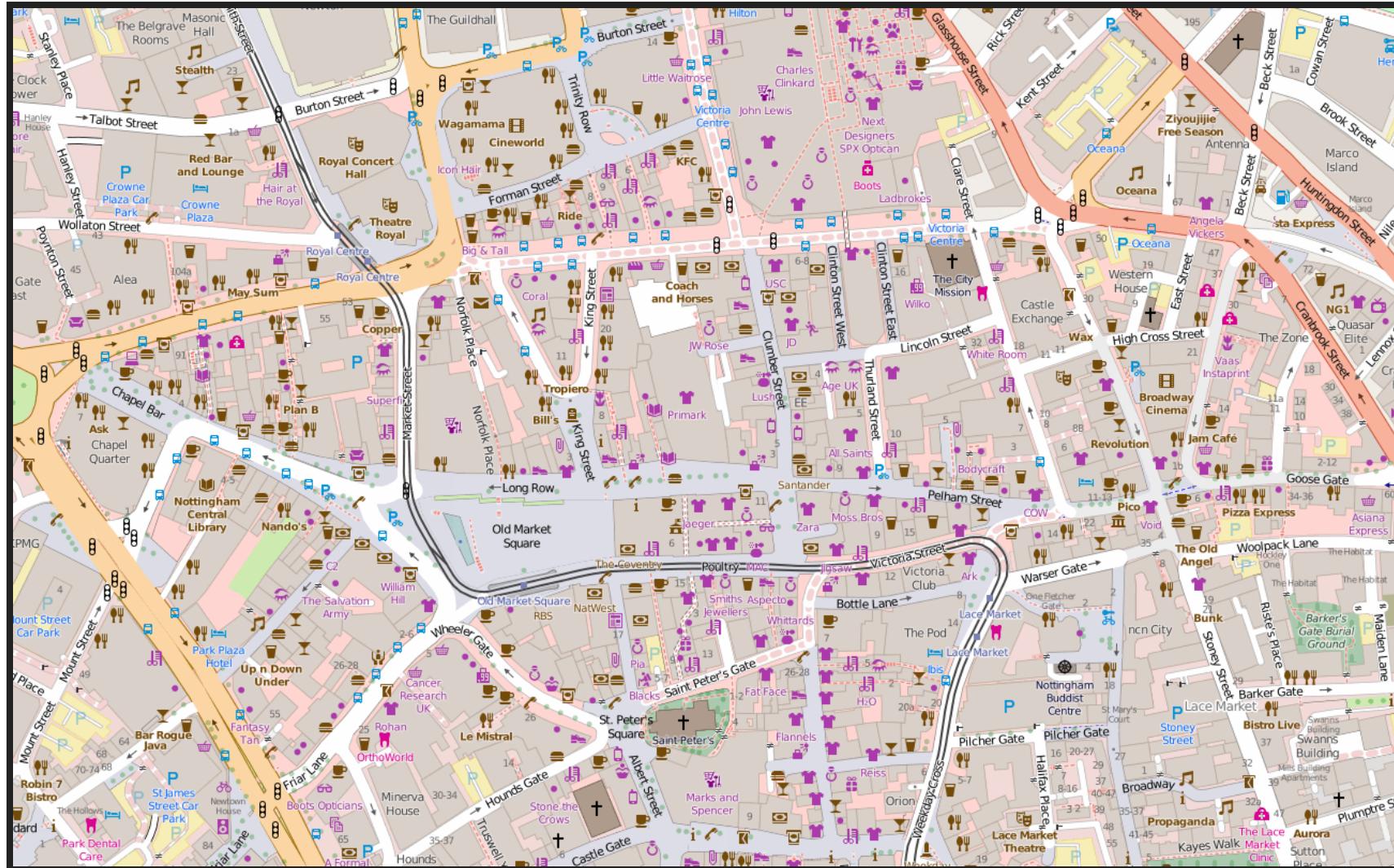
WICHTIG FÜR FEEDBACK

- Detailtiefe
- Darstellung von vielen Features
- Kontrast zwischen Elementen zur Unterscheidung
 - visuelle Ästhetik eher untergeordnet

FEEDBACK-MECHANISMS UND SEINE AUSWIRKUNGEN AUF OSM-CARTO



von [Imagico](#) in #1991 (comment)



17/52.95392/-1.14713 (© OpenStreetMap contributors)

~ 70 offene Anfragen für die Darstellung neuer Features

Der Vorschlag eines neuen Features ist der häufigste Grund
für ein neues Ticket.

VISUELLE ÄSTHETIK?

Cartographers make aesthetic judgments when designing maps to ensure that the content forms a clear expression of the theme(s).

[...] aesthetic judgments [...] are fundamental to the cartographer's symbolisation and as such are integral to the function of maps.

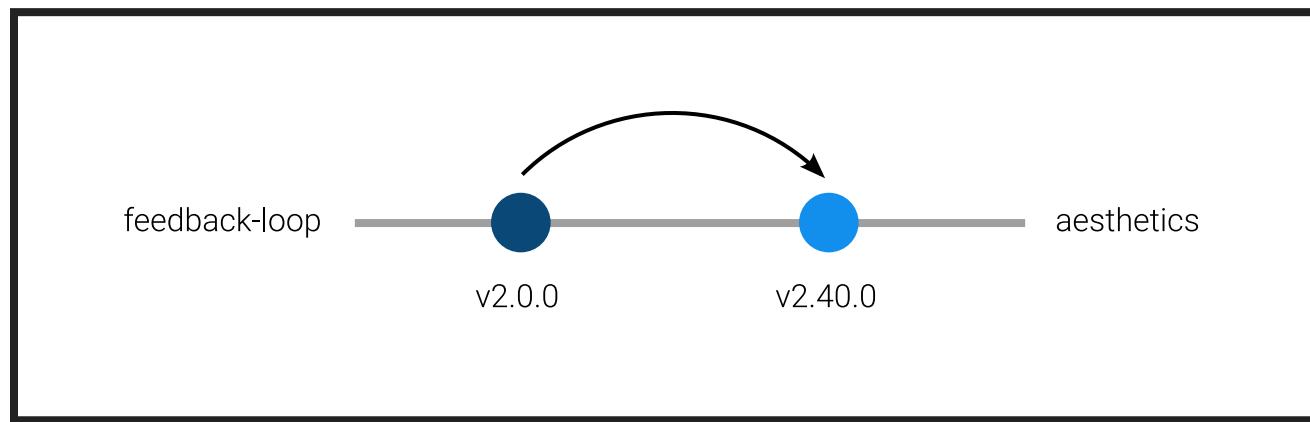
VISUELLE ÄSTHETIK IN OSM-CARTO

The colour palette should be moving towards pastel/light/desaturated for background layers, midtones for streets and save highlights/bolds/saturated for points of interest.

aus den [Cartography guidelines](#)

~ letzte 2 Jahre der Entwicklung standen im Zeichen von
Konsolidierung und Überarbeitung

neue Features bildeten eher die Ausnahme, außer in letzter
Zeit bei POIs

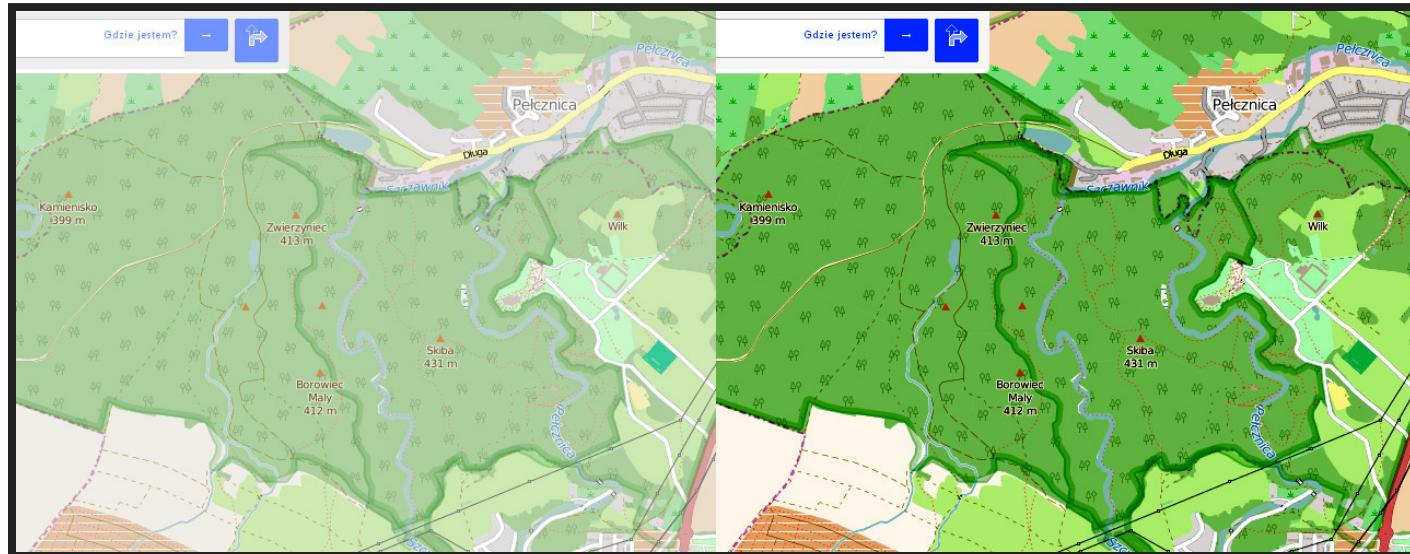


REAKTIONEN MANCHER BENUTZER

"Usability: New map openstreetmap-carto is too pale"
(#1863)

"Meiner Meinung nach deutlich zu wenig Kontrast."
(deutsches Forum)

VORSCHLÄGE BZW. VORSTELLUNGEN



#1863 (comment) cf. polnisches Forum

WARUM GIBT ES NICHT MEHR STILE (AUF OSM.ORG)?

- Erstellungs- und Betreuungsaufwand relativ hoch
- Kosten und Aufwand für Betrieb relativ hoch (siehe [Guidelines for new tile layers](#))

IM BESONDEREN:

- Global scope and coverage
- Capable of meeting traffic demands
- Up-to-date data

VECTOR TILES^{*} UND IN-BROWSER RENDERING?

* Markus Roth: [Vector Tiles from OpenStreetMap](#), 5.7. 09:30 GI-Studio

- separiert Daten- und Layerdefinition und Styling
- **Was** kann gerendert werden vs. **wie** wird es gerendert
- Stil wird im Browser gerendert ohne zusätzliche notwendige Infrastruktur
- würde die Anzahl verfügbarer Stile wohl drastisch erhöhen

AUSWIRKUNGEN

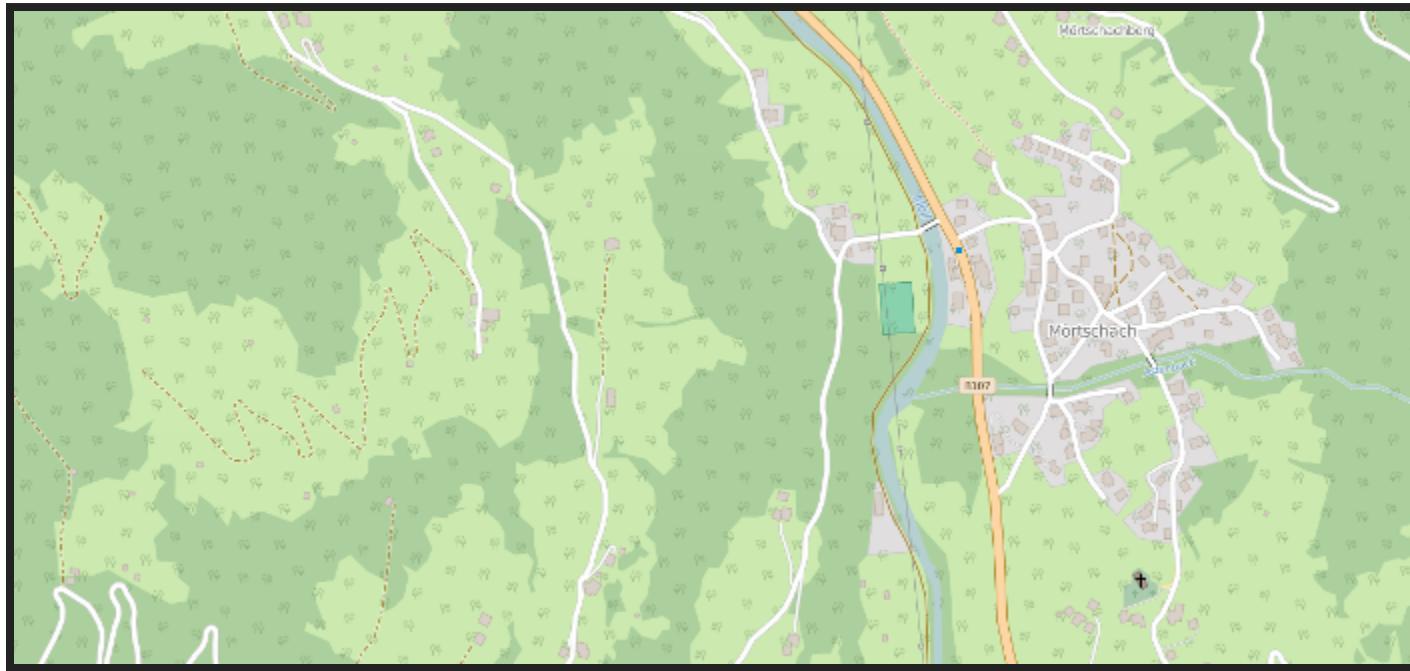
- Stile würde alle eine gewisse Ähnlichkeit aufweisen
- Grundsatzdiskussionen und Feature-Wünsche würden nicht verschwinden
- Repository für Vector Tile-Definition würde zu neuem osm-carto werden
- höherer Aufwand und Datenvolumen für OSMF Server (?)

WICHTIGE STEUERUNGSFUNKTION DATENQUALITÄT UND VERMEIDUNG VON TAGGEN FÜR DEN RENDERER

DATENQUALITÄT

Fehler in den Daten sollen nicht "korrigiert" werden (siehe
[Cartography guidelines](#))

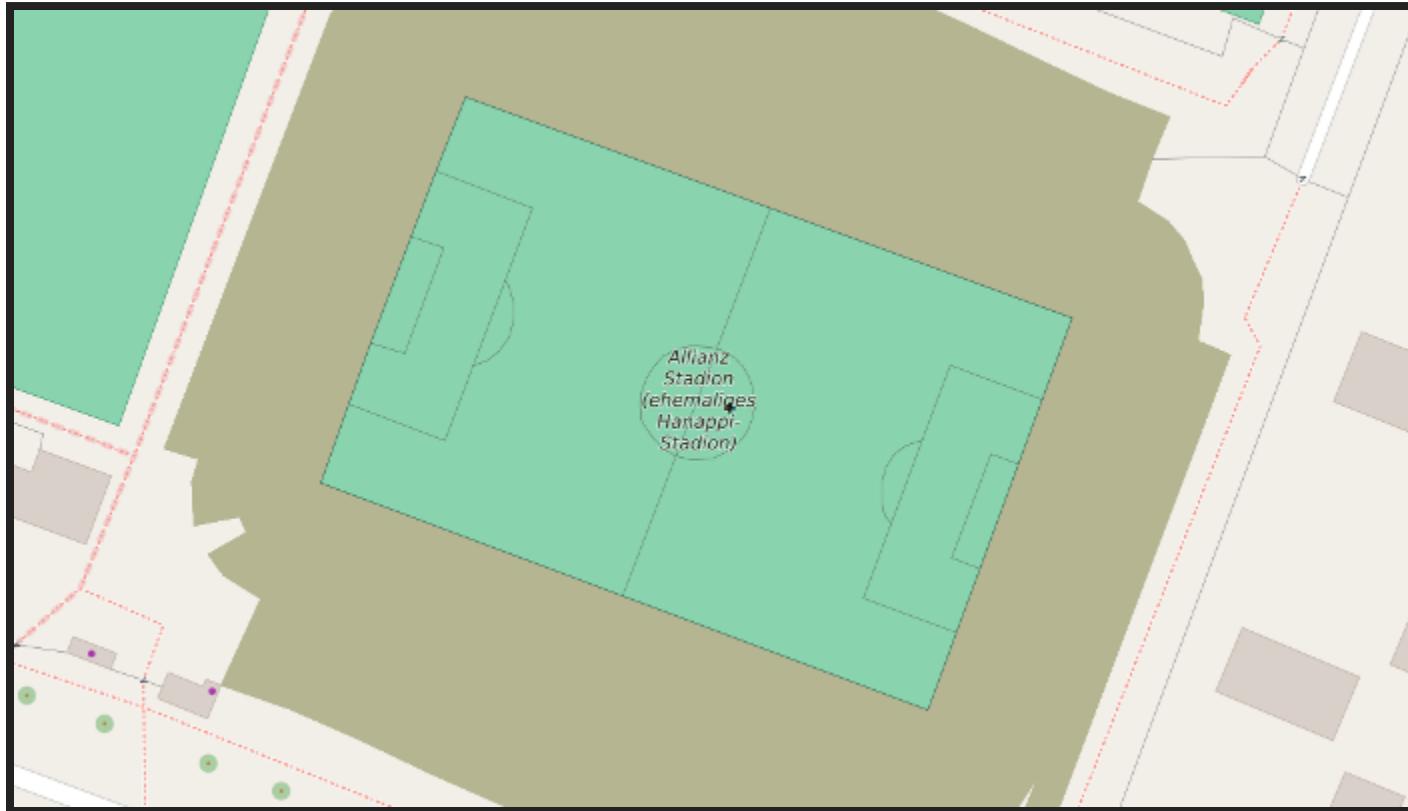
Beispiel: Wald als "Hintergrund"



vor #1242/#1728 war es viel schwieriger solche Fälle zu erkennen

durch das eher hässliche Aussehen wird die Korrektur wohl beschleunigt

TAGGEN FÜR DEN RENDERER



Sportfeldmarkierung als barrier=line

Ursache: jede Linie mit `barrier=*` wurde gleich dargestellt

Bild oben ist nettes Easter Egg, aber Darstellung begünstigt Inflation/Aufsplitterung bei Barrier-Werten

Änderung in [#1986](#) auf etablierte und gut definierte Werte mit Hilfe von Taginfo und Wiki

KLEINE ENTWICKLERGEMEINDE BEI OSM-CARTO

- 4 Maintainer (entscheiden was geändert wird und was nicht)
- 2015: 425 Commits
 - 14 externe Beitragende
 - 8 davon mit mehr als 1 Commit
 - 4 davon mit mehr als 10 Commits

[Statistik auf GitHub](#)

ZUM VERGLEICH: ID

- ~ 3 Maintainer (unterstützt von Mapbox?)
- 2015: 488 Commits
 - 34 externe Beitragende
 - 15 davon mit mehr als 1 Commit
 - 3 davon mit mehr als 10 Commits

[Statistik auf GitHub](#)

ANATOMIE EINER ÄNDERUNG

Annahme: Entwicklungsumgebung bereits vorhanden

ÄNDERUNG DER FARBE VON STATIONSBESCHRIFTUNGEN



#1910

Beobachtung: Farbe bei Stationsbeschriftungen ist unschön und oft schwer lesbar

Experimentieren mit dunkleren Farben, Halos und Schriftschnitt in Kosmtik

```
@station-color: #7981b0;  
@station-text: darken(saturate(@station-color, 15%), 10%);  
...  
[zoom >= 14] {  
    text-name: "[name]";  
    text-face-name: @bold-fonts;  
    text-size: 9;  
    text-fill: @station-text;  
    text-dy: 8;  
    text-halo-radius: 1.5;  
    text-halo-fill: rgba(255,255,255,0.6);  
    text-wrap-width: 0;  
    text-placement: interior;  
}
```

in [stations.mss](#)

1. Überlegen welche Kategorien für Test sinnvoll sind, hier:
Stadt/Land, U-Bahn, Tram, Bahnhöfe und kleinere
Stationshalte
2. Suchen von Testregionen [opt. zurück zu 1.]
3. Anfertigen von Testbildern (vorher/nachher)
4. Einbringen und Dokumentieren der Änderung auf GitHub
(Pull Request)
5. Diskussion, weitere Aspekte werden eingebracht,
mögliche Änderungswünsche
6. Durchführen etwaiger Änderungen [zurück zu 5.]
7. Maintainer-Review
8. Aufnahme oder Ablehnung

ZUSAMMENFASSUNG

- osm-carto sitzt zwischen den Stühlen Feedback-Loop und Ästhetik
- wichtige Steuerungsfunktion und große Verantwortung
- sehr wenig Beitragende
- die Funktion von osm-carto ist zu zentral, OSM braucht mehr Karten (auf der Startseite)

ENTWICKLERSETUP

- Daten: PostgreSQL + PostGIS, osm2pgsql für Import; und Datenextrakt
- Code: openstreetmap-carto (+ Git, Python)
- Editor: Kosmtik (+ Node.js)

VORBEREITUNG

- PostgreSQL installieren
- Git installieren
- Node.js installieren
- evtl. Python installieren (auf MacOS X und Linux bereits vorhanden)
- osm2pgsql installieren
- Datenextract besorgen: z.B. Geofabrik oder Mapzen Metro-Extracts

OPENSTREETMAP-CARTO

- Ordner für openstreetmap-carto anlegen und per Kommandozeile (oder Git-Bash) in den Ordner wechseln
- Im [GitHub-Repository](#) auf "Clone or download" und dann auf "Use HTTPS" gehen
- URL kopieren
- `git clone <kopierte URL> .` ausführen (Code wird im aktuellen Verzeichnis erstellt)
- im Verzeichnis `./get-shapefiles.sh` bzw. `./get-shapefiles.py` (was dann existiert) ausführen

DATENBANK

- Datenbank anlegen (Name z.B. osm)
- PostGIS aktivieren: CREATE EXTENSION postgis;
- opt. HStore aktivieren: CREATE EXTENSION hstore;
- Daten importieren (Linux):

```
sudo -u postgres osm2pgsql --create --slim --cache CACHE --number-proc
```

CACHE = Größe RAM - belegter Speicher - X (nicht zu viel, da sonst Swap notwendig)

NUM = Anzahl Prozessorkerne

Der Import kann je nach Extrakt-Größe einige Zeit dauern!

KOSMTIK INSTALLIEREN

- Im [GitHub-Repository](#) auf "Clone or download" und dann auf "Download ZIP" gehen (bzw. Methode wie für osmcarto oben anwenden)
- ZIP-Datei in Ordner entpacken
- Im Ordner `npm install` ausführen (Node.js notwendig)

KOSMTIK KONFIGURIEREN UND STARTEN

- Local-Config Datei anlegen, z.B. osm.localconfig:

```
exports.LocalConfig = function (localizer, project) {
  localizer.where('center').then([13.0463, 47.7978, 15]); // Salzburg
  localizer.where('Layer').if({ 'Datasource.type': 'postgis' }).then({
    "Datasource.dbname": "osm",
    "Datasource.password": "postgres",
    "Datasource.user": "postgres",
    "Datasource.host": "localhost"
  });
  // You can also do it in pure JS
  project.mml.bounds = [1, 2, 3, 4];
}
```

- Kosmtik starten:

```
node index.js serve --localconfig path/to/osm.localconfig path/to/ope
```

OSM-CARTO BEARBEITEN

- eigenen Texteditor verwenden z.B. Atom, gedit, TextWrangler, Notepad++
- Bearbeitungen für den Stil in den Dateien *.mss vornehmen
- Änderungen in Kosmtik begutachten
- Bearbeitungen für die Layer-Definitionen in der Datei project.yaml vornehmen (danach ./yaml2mml.py ausführen)