Neues von MapProxy

Neue (und alte) Funktionen in der Praxis

FOSSGIS 2016, Salzbug

Oliver Tonnhofer <olt@omniscale.de | @oltonn>

Omniscale

Über mich

Oliver Tonnhofer

- Omniscale GmbH & Co. KG, Oldenburg
  - Open Source Entwicklung (Client/Server)
  - MapProxy und Imposm Entwicklung und Support
  - OpenStreetMap Hosting

Omniscale

MapProxy in der Praxis

Omniscale

MapProxy in der Praxis

- Wer kennt MapProxy?

Omniscale

MapProxy in der Praxis

- Wer kennt MapProxy?
- Wer nutzt MapProxy?

Omniscale

Inhalt

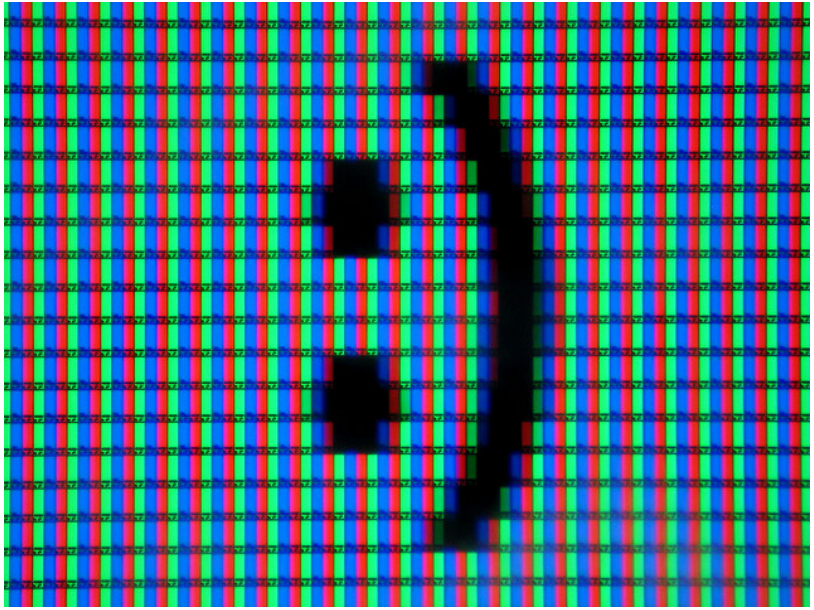
- Mobile Anwendungen
- Arbeiten mit bestehenden Kacheln
- Nachträgliche Bildbearbeitung
- Absicherung von Diensten
- Effizientes Seeding

Omniscale


Omniscale

Mobile Anwendungen

Omniscale



Bildpixel = Displaypixel

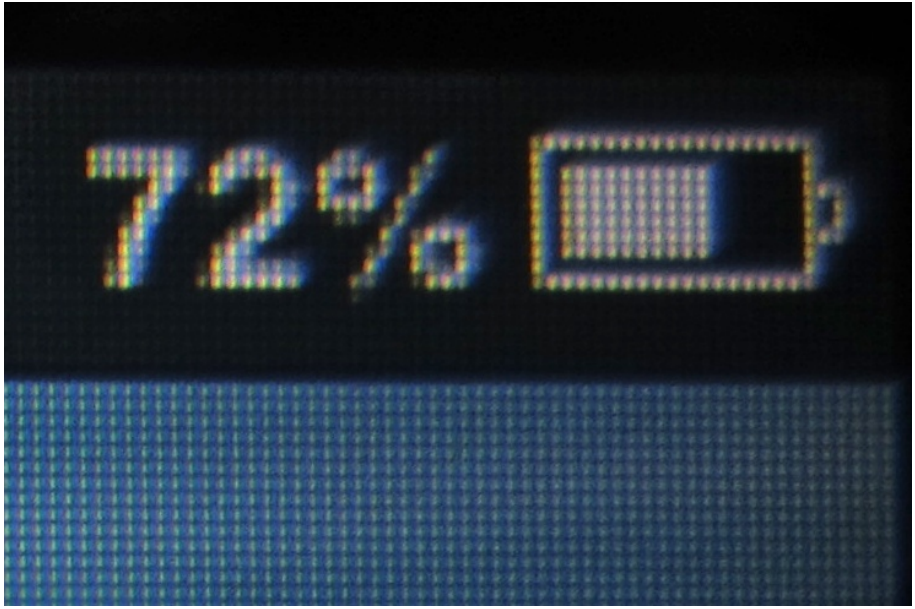
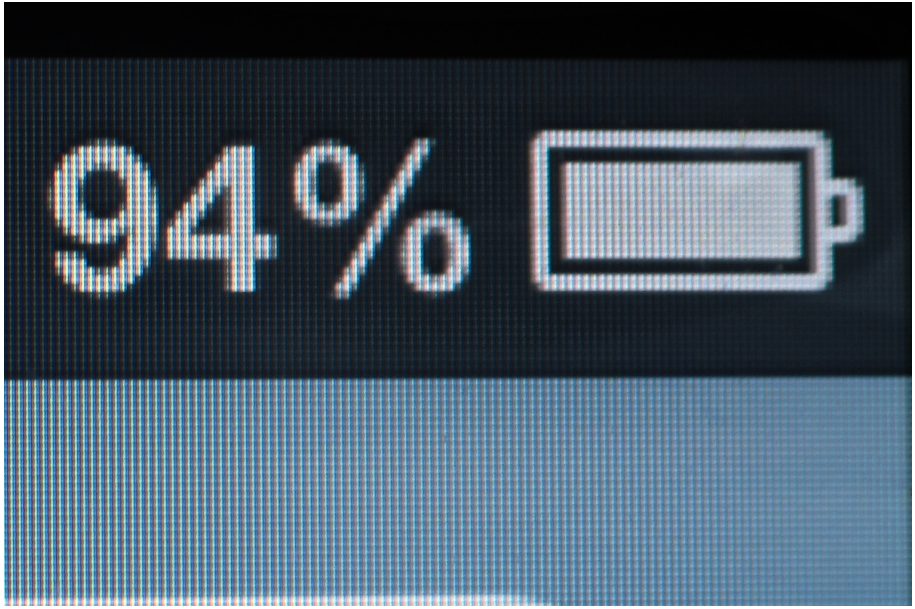


Bild: Public Domain



Retina-, HQ-, HightDPI- Displays

Bild: Public Domain

Omniscale

Bildpixel = Displaypixel

Bild: Public Domain, Natural Earth Data

Omniscale

Bildpixel = Displaypixel

Bildpixel = 2 × Displaypixel

Bild: Public Domain, Natural Earth Data

Omniscale

2 × Bildpixel = 2 × Displaypixel

↓

Bild: Public Domain, Natural Earth Data

Omniscale

Source mit vergrößerten Schriften

Seit MapProxy 1.8.0

Mapnik mit scale_factor

```
sources:
  map_hq_source:
    type: mapnik
    mapfile: ./mapnik.xml
    scale_factor: 2
```

Omniscale

Source mit vergrößerten Schriften

MapServer mit DPI

```
sources:
  map_hq_source:
    type: wms
    req:
      url: http://example.org/mapserv
      layers: basemap
      map_resolution: 144 # 2 * DEFRESOLUTION
```

Omniscale

Source mit vergrößerten Schriften

MapServer mit DPI

```
sources:
  map_hq_source:
    type: wms
    req:
      url: http://example.org/mapserv
      layers: basemap
      map_resolution: 144 # 2 * DEFRESOLUTION
```

Andere Server: ggf. separater Style

Grid mit 512x512 Pixel

```
grids:
  webmercator:
    srs: "EPSG:3857"
    origin: nw
    min_res: 156543.03392804097
  webmercator_hq:
    srs: "EPSG:3857"
    origin: nw
    min_res: 78271.51696402048
    tile_size: [512, 512]
```

Omniscale

Zwei Caches

```
caches:
  map_cache:
    grids: [webmercator]
    sources: [map_source]
  map_hq_cache:
    grids: [webmercator_hq]
    sources: [map_hq_source]
```

Omniscale

Layer mit zwei `tile_sources`

Seit MapProxy 1.8.2

```
layers:
  - name: map
    title: My Map
    tile_sources: [map_cache, map_hq_cache]
```

Omniscale

Clientseitig

```
var tileMatrix = 'webmercator';
if (window.devicePixelRatio > 1) {
  tileMatrix = 'webmercator_hq';
}
```

Omniscale

Clientseitig

```
var tileMatrix = 'webmercator';
if (window.devicePixelRatio > 1) {
  tileMatrix = 'webmercator_hq';
}
```

OpenLayers 3

```
var hiDPI = ol.has.DEVICE_PIXEL_RATIO > 1;
var layer = hiDPI ? 'webmercator' : 'webmercator_hq';
var tilePixelRatio = hiDPI ? 2 : 1;

map.addLayer(new ol.layer.Tile({
  source: new ol.source.WMTS({
    ...
    tilePixelRatio: tilePixelRatio,
  })
});
```

Omniscale

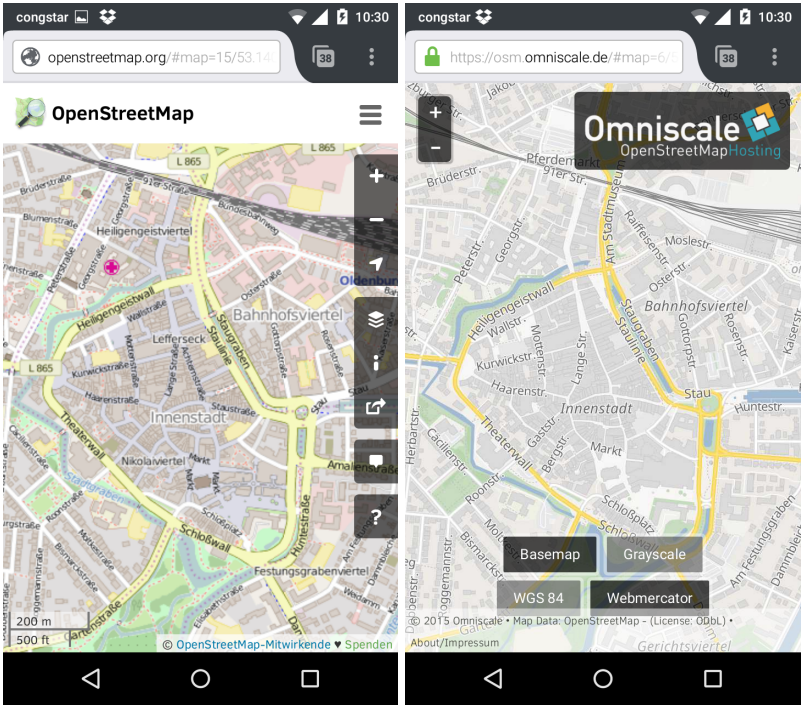


Bild: © 2016 Omniscale • OpenStreetMap - contributors

Omniscale

Größere Bilder = mehr Bandbreite?

Omniscale

Größere Bilder = mehr Bandbreite?

Zwei Caches mit separaten Optionen

```
caches:
  map_cache:
    grids: [webmercator]
    sources: [map_source]
  map_hq_cache:
    grids: [webmercator_hq]
    sources: [map_hq_source]
    format: image/jpeg
    image:
      encoding_options:
        jpeg_quality: 70
```

Omniscale


Omniscale

Arbeiten mit bestehenden Kacheln

Omniscale

Bestehende Kachel**dienste** einbinden

```
sources:
  my_tile_source:
    type: tile
    grid: GLOBAL_WEBMERCATOR
    url: http://my-tile-server/%(z)s/%(x)s/%(y)s.png
```

Omniscale

Bestehende Kachel**dienste** einbinden

```
sources:
  my_tile_source:
    type: tile
    grid: GLOBAL_WEBMERCATOR
    url: http://my-tile-server/%(z)s/%(x)s/%(y)s.png
```

URL anpassbar:

```
url: http://my-tile-server/service?zoom=%(z)s&col=%(x)s&row=%(y)s&format=png
```

- x, y, z
- bbox
- arcgiscache_path
- tms_path
- quadkey

Nutzungsbedingungen beachten!

- OSM: http://wiki.openstreetmap.org/wiki/Tile_usage_policy
- Bing und Google Maps: Gestatten nur Zugriff über Webbrowser
- Andere Anbieter: ???

Omniscale

Bestehende Kachel**dateien** einbinden

```
caches:
  file_cache:
    sources: []
    grids: [GLOBAL_WEBMERCATOR]
    cache:
      type: file
      directory_layout: tilecache
      directory: /path/to/tiles
```

Omniscale

Bestehende Kachel**dateien** einbinden

```
caches:
  file_cache:
    sources: []
    grids: [GLOBAL_WEBMERCATOR]
    cache:
      type: file
      directory_layout: tilecache
      directory: /path/to/tiles
```

Format anpassbar:

- tms: 04/051/123.png
- arcgis: L04/R00000051/C00000123.png

Einbinden von MBTiles

Erstellt mit gdal2tiles, MapTiler, etc.

```
caches:
  mbtiles_cache:
    sources: []
    grids: [GLOBAL_MERCATOR]
    cache:
      type: mbtiles
      filename: /path/to/my.mbtiles
```

Einbinden von MBTiles

Erstellt mit gdal2tiles, MapTiler, etc.

```
caches:
  mbtiles_cache:
    sources: []
    grids: [GLOBAL_MERCATOR]
    cache:
      type: mbtiles
      filename: /path/to/my.mbtiles
```

Achtung: GLOBAL_MERCATOR statt GLOBAL_WEBMERCATOR, da MBTiles eine andere Zählweise der Kacheln verwendet

Kacheln umprojizieren

Direkt per WMS

```
services:
  wms:
    srs: ['EPSG:25832']

layers:
  - name: osm
    title: OSM
    sources: [osm_cache]
```

Omniscale

Dienst auf BBOX beschränken

Mit coverage für die Quelle

```
sources:
  osm_source:
    type: tile
    grid: GLOBAL_WEBMERCATOR
    url: http://my-tile-server/%(z)s/%(x)s/%(y)s.png
    coverage:
      bbox: [4, 46, 16, 56]
      srs: 'EPSG:4326'
```

Omniscale

Dienst auf BBOX beschränken

Seit MapProxy 1.7.0

Oder für den gesamten Dienst

```
services:
  wms:
    srs: ['EPSG:25832']
    bbox_srs:
      - srs: 'EPSG:25832'
        bbox: [12855, 5094533, 1041978, 6228213]
```

Omniscale

Webmercator → UTM

Bild: © OpenStreetMap contributors

Omniscale

Cache umprojizieren

Grid in Ziel-Projektion

```
grids:
  utm32_adv:
    bbox: [-46133.17, 5048875.26857567, 1206211.10142433, 6301219.54]
    srs: 'EPSG:25832'
    origin: ul
    min_res: 4891.96981025128
    num_levels: 15
```

Omniscale

Eingangs- und Ziel-Cache

```
caches:
  osm_cache:
    grids: [GLOBAL_WEBMERCATOR]
    sources: [osm_source]

  osm_utm32_cache:
    grids: [utm32_adv]
    sources: [osm_cache]
```

Omniscale

Zwei Caches

`osm_cache_EPSG3857/07/000/000/067/000/000/043.png`

`osm_utm32_cache_EPSG25832/02/000/000/001/000/000/002.png`

Omniscale

Temporärer Eingangs-Cache

```
caches:
  osm_cache:
    grids: [GLOBAL_WEBMERCATOR]
    disable_storage: true
    sources: [osm_source]
```

Omniscale

Umprojizieren

- Immer mit Qualitätseinbußen. Insbesondere:
  - bei unterschiedliche Auflösungen (Zoom-Stufen)
  - von/nach EPSG:4326/WGS84
  - bei Beschriftungen

Omniscale

Umprojizieren

- Immer mit Qualitätseinbußen. Insbesondere:
  - bei unterschiedliche Auflösungen (Zoom-Stufen)
  - von/nach EPSG:4326/WGS84
  - bei Beschriftungen
- (Etwas) Bessere Ergebnisse mit:

```
resampling_method: bicubic
```

Omniscale

Umprojizieren

- Immer mit Qualitätseinbußen. Insbesondere:
  - bei unterschiedliche Auflösungen (Zoom-Stufen)
  - von/nach EPSG:4326/WGS84
  - bei Beschriftungen
- (Etwas) Bessere Ergebnisse mit:

```
resampling_method: bicubic
```

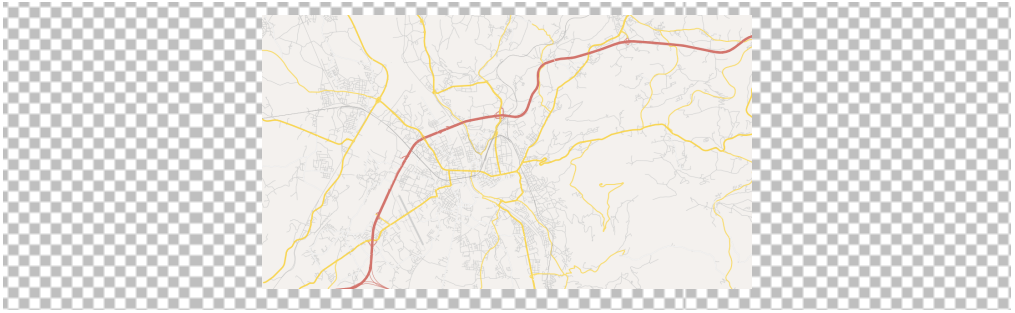
- Beste Ergebnisse: Nur mit Diensten, die die gewünschte Projektion unterstützen.


Omniscale

Nachträgliche Bildbearbeitung

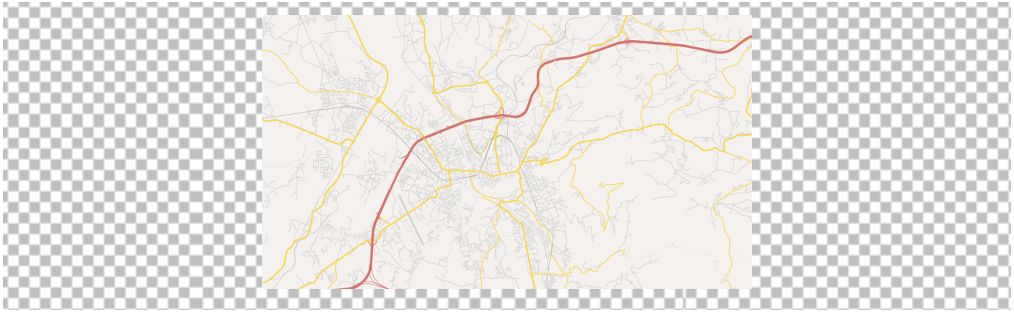
Omniscale

Karten ohne Transparenz



Omniscale

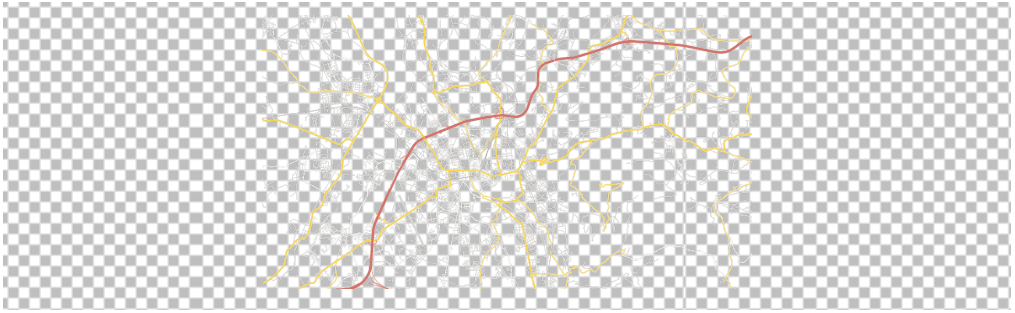
Karten ohne Transparenz



```
source:
  wms_source:
    type: wms
    req:
      url: http://example.org/service?
      layers: layer-ohne-transparenz
    image:
      transparent_color: '#F2F2EF'
      transparent_color_tolerance: 5
```

Omniscale

Karten ohne Transparenz



```
source:
  wms_source:
    type: wms
    req:
      url: http://example.org/service?
      layers: layer-ohne-transparenz
    image:
      transparent_color: '#F2F2EF'
      transparent_color_tolerance: 5
```

Omniscale

Nachträgliche Bildbearbeitung

Decorate Image API

- Für Wasserzeichen, Logos, etc.

Omniscale

Nachträgliche Bildbearbeitung

Decorate Image API

- Für Wasserzeichen, Logos, etc.
- Nachträgliches Bearbeiten der MapProxy Antwort
- Eigener Python Programmcode
- Python Image Library (PIL/Pillow)

Omniscale

Hello Decorate API

```
def annotate_img(image, service, layers, environ, query_extent, **kw):
    img = image.as_image().convert('RGBA')

    draw = ImageDraw.Draw(img)
    font = ImageFont.truetype("/Library/Fonts/DejaVuSans.ttf", 120)
    fill = ImageColor.getrgb('black')

    draw.text((100, 100), "Hello!", font=font, fill=fill)

    return ImageSource(img, image.image_opts)
```

Omniscale

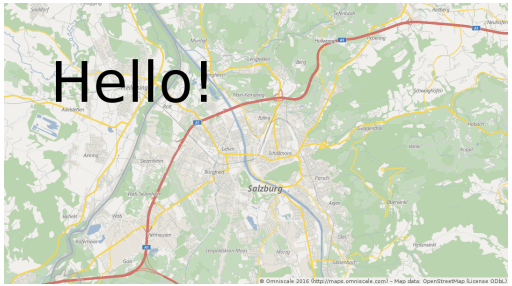
Hello Decorate API

```
def annotate_img(image, service, layers, environ, query_extent, **kw):
    img = image.as_image().convert('RGBA')

    draw = ImageDraw.Draw(img)
    font = ImageFont.truetype("/Library/Fonts/DejaVuSans.ttf", 120)
    fill = ImageColor.getrgb('black')

    draw.text((100, 100), "Hello!", font=font, fill=fill)

    return ImageSource(img, image.image_opts)
```



Omniscale

Antworten abhängig von der Anfrage

- Zugriff auf
  - Anfragetyp (WMS, WMTS, TMS, etc.)
  - Ausdehnung
  - Angefragte Layer
  - Client (IP, User-Agent, etc)

Omniscale

Antworten abhängig von der Anfrage

```
def annotate_img(image, service, layers, environ, query_extent, **kw):
    msg = "Hello"
    if 'osm' in layers:
      msg = "Hello OSM"

    img = image.as_image().convert('RGBA')

    draw = ImageDraw.Draw(img)
    font = ImageFont.truetype("/Library/Fonts/DejaVuSans.ttf", 120)
    fill = ImageColor.getrgb('black')

    draw.text((100, 100), msg, font=font, fill=fill)

    return ImageSource(img, image.image_opts)
```

Weitere Dokumentation: https://mapproxy.org/docs/nightly/decorate_img.html

Nachträgliche Bildbearbeitung

Seit MapProxy 1.9.0

Kanalkombination

Omniscale

Nachträgliche Bildbearbeitung

Seit MapProxy 1.9.0

Kanalkombination

Statt Sourcen pro Cache:

```
caches:
 my_cache:
     sources: [wms_source]
```

Omniscale

Nachträgliche Bildbearbeitung

Seit MapProxy 1.9.0

Kanalkombination

Statt Sourcen pro Cache:

```
caches:
 my_cache:
     sources: [wms_source]
```

Sourcen pro Farbkanal:

```
caches:
 my_cache:
     sources:
         r: [{source: wms_source, band: 0}]
         g: [{source: wms_source, band: 1}]
         b: [{source: wms_source, band: 2}]
```

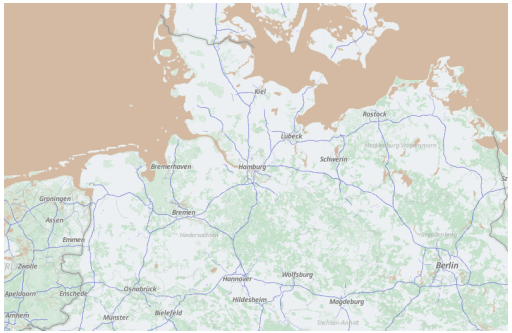
Vertauschen der Farbkanäle

```
caches:
 color_cache:
     sources:
         r: [{source: wms_source, band: 2}]
         g: [{source: wms_source, band: 1}]
         b: [{source: wms_source, band: 0}]
```

Omniscale

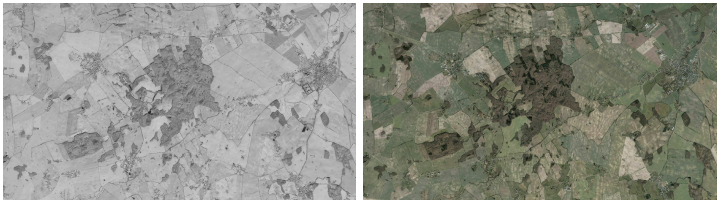
Vertauschen der Farbkanäle

```
caches:
 color_cache:
     sources:
         r: [{source: wms_source, band: 2}]
         g: [{source: wms_source, band: 1}]
         b: [{source: wms_source, band: 0}]
```

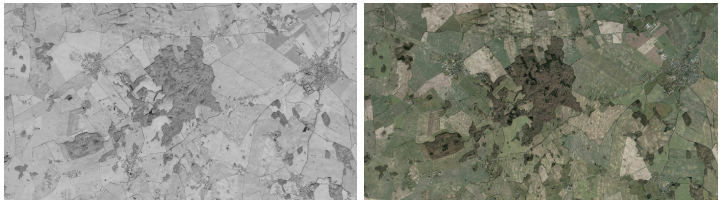


Omniscale

Falschfarbenbilder

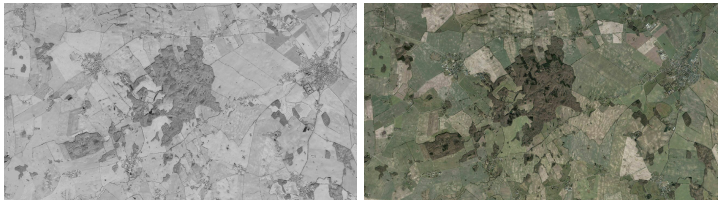


Falschfarbenbilder

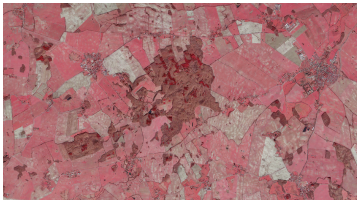


```
caches:
 cir_cache:
    sources:
      r: [{source: dop_ir, band: 0}]
      g: [{source: dop_rgb, band: 0}]
      b: [{source: dop_rgb, band: 1}]
```

Falschfarbenbilder



```
caches:
 cir_cache:
    sources:
      r: [{source: dop_ir, band: 0}]
      g: [{source: dop_rgb, band: 0}]
      b: [{source: dop_rgb, band: 1}]
```

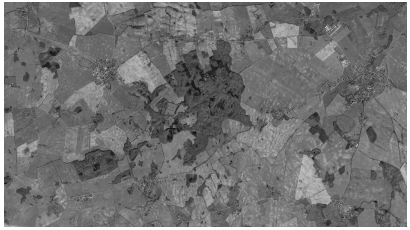


Graustufenbilder

```
caches:
 grayscale_cache:
    sources:
      l: [
        {source: dop_rgb, band: 0, factor: 0.25},
        {source: dop_rgb, band: 1, factor: 0.7},
        {source: dop_rgb, band: 2, factor: 0.05},
      ]
```

Graustufenbilder

```
caches:
 grayscale_cache:
    sources:
      l: [
        {source: dop_rgb, band: 0, factor: 0.25},
        {source: dop_rgb, band: 1, factor: 0.7},
        {source: dop_rgb, band: 2, factor: 0.05},
      ]
```



Ohne dauerhaftes Caching

```
caches:
 grayscale_cache:
    format: image/jpeg
    disable_storage: true
    meta_size: [1, 1]
    meta_buffer: 0
    sources:
      l: [
        {source: dop_rgb, band: 0, factor: 0.25},
        {source: dop_rgb, band: 1, factor: 0.7},
        {source: dop_rgb, band: 2, factor: 0.05},
      ]
```

Absicherung von Diensten

Omniscale

Absichern von Diensten

Security API

- Ähnlich wie Decorate Image API
- Eigener Python Programmcode
  - zum Erkennen von Benutzern und Gruppen
  - zum Bestimmen der Berechtigung
- MapProxy setzt Berechtigungen um

Omniscale

Alles erlauben

```
def authorize(service, environ, layers=[], **kw):
    return {'authorized': 'full'}
```

Omniscale

Ein Layer nur für WMTS verfügbar machen

```
def authorize(service, environ, layers=[], **kw):
    if 'tileonly' in layers and not service.startswith('wmts'):
        return {'authorized': 'none'}
    else:
        return {'authorized': 'full'}
```

Omniscale

Einzelne Nutzer einschränken

```
def authorize(service, environ, layers=[], **kw):
   if environ['REMOTE_USER'] == 'bob':
    return {
      'authorized': 'partial',
      'layers': {
        'sat': {'map': True},
        'osm': {
            'map': True,
            'limited_to': {
              'geometry': Point(8, 53).buffer(2),
              'srs': 'EPSG:4326',
            }
    ...
```

Omniscale

Einzelne Nutzer einschränken

```
def authorize(service, environ, layers=[], **kw):
   if environ['REMOTE_USER'] == 'bob':
    return {
      'authorized': 'partial',
      'layers': {
        'sat': {'map': True},
        'osm': {
            'map': True,
            'limited_to': {
              'geometry': Point(8, 53).buffer(2),
              'srs': 'EPSG:4326',
            }
    ...
```

Bild: © OpenStreetMap - contributors; Public Domain NASA

Omniscale

Beschränkung mit komplexen Geometrien

Bild: © OpenStreetMap - contributors; Public Domain NASA

Omniscale

Beschränkung mit komplexen Geometrien

Durch Python sind die Möglichkeiten (nahezu) unbegrenzt.

Auch für WMTS/TMS

Mehr Informationen

- Dokumentation: https://mapproxy.org/docs/nightly/auth.html
- FOSSGIS 2013: https://talks.omniscale.de/2013/fossgis/mapproxy_security.pdf


Omniscale

Effizientes Seeding

Omniscale

Einfaches Seeding

```
seeds:
  coverage_seed:
    caches: [osm_cache]
    levels:
      to: 11
```

Omniscale

Wie viele Kacheln?

```
% mapproxy-util grids mapproxy.yaml
utm32_adv:
    Configuration:
        bbox: [-46133.17, 5048875.26857567, 1206211.10142433, 6301219.54]
        min_res: 4891.96981025128
        num_levels: 15
        origin: 'ul'
        srs: 'EPSG:25832'
        tile_size*: [256, 256]
    Levels: Resolutions, # x * y = total tiles
        00:  4891.96981025128,    #      1 * 1      =          1
        01:  2445.98490512564,    #      2 * 2      =          4
        02:  1222.99245256282,    #      4 * 4      =         16
        03:  611.49622628141,     #      8 * 8      =         64
        04:  305.748113140705,    #     16 * 16     =        256
        05:  152.8740565703525,   #     32 * 32     =       1024
        06:  76.43702828517625,   #     64 * 64     =       4096
        07:  38.21851414258813,   #    128 * 128    =      16384
        08:  19.109257071294063,  #    256 * 256    =      65536
        09:  9.554628535647032,   #    512 * 512    =     262144
        10:  4.777314267823516,   #   1024 * 1024   =      1.05M
        11:  2.388657133911758,   #   2048 * 2048   =      4.19M
        12:  1.194328566955879,   #   4096 * 4096   =     16.78M
        13:  0.5971642834779395,  #   8192 * 8192   =     67.11M
        14:  0.29858214173896974, #  16384 * 16384  =    268.44M
```

Omniscale

Seeding mit Coverages einschränken

Omniscale

Seeding mit Coverages einschränken

Beschränken auf BBOX

```
coverages:
  germany:
     bbox: [5, 50, 10, 55]
     srs: 'EPSG:4326'
```

Omniscale

Seeding mit Coverages einschränken

Beschränken auf BBOX

```
coverages:
  germany:
     bbox: [5, 50, 10, 55]
     srs: 'EPSG:4326'
```

```
seeds:
  coverage_seed:
    coverages: [germany]
    caches: [osm_cache]
    levels:
      to: 11
```

Omniscale

Wie viele Kacheln?

```
% mapproxy-util grids mapproxy.yaml --seed seed.yaml --coverage germany_bbox
utm32_adv:
    Configuration:
        bbox: [-46133.17, 5048875.26857567, 1206211.10142433, 6301219.54]
        min_res: 4891.96981025128
        num_levels: 15
        origin: 'ul'
        srs: 'EPSG:25832'
        tile_size: [256, 256]
    Coverage: germany_bbox covers approx. 54.5032% of the grid BBOX
    Levels: Resolutions, # x * y = total tiles (approx. tiles within coverage)
        00:  4891.96981025128,    #      1 * 1      =          1 (0)
        01:  2445.98490512564,    #      2 * 2      =          4 (2)
        02:  1222.99245256282,    #      4 * 4      =         16 (8)
        03:  611.49622628141,     #      8 * 8      =         64 (34)
        04:  305.748113140705,    #     16 * 16     =        256 (139)
        05:  152.8740565703525,   #     32 * 32     =       1024 (558)
        06:  76.43702828517625,   #     64 * 64     =       4096 (2232)
        07:  38.21851414258813,   #    128 * 128    =      16384 (8929)
        08:  19.109257071294063,  #    256 * 256    =      65536 (35719)
        09:  9.554628535647032,   #    512 * 512    =     262144 (142876)
        10:  4.777314267823516,   #   1024 * 1024   =      1.05M (571507)
        11:  2.388657133911758,   #   2048 * 2048   =      4.19M (   2.29M)
        12:  1.194328566955879,   #   4096 * 4096   =     16.78M (   9.14M)
        13:  0.5971642834779395,  #   8192 * 8192   =     67.11M (  36.58M)
        14:  0.29858214173896974, #  16384 * 16384  =    268.44M ( 146.31M)
```

Omniscale

Reicht eine Bounding Box?

Colorado

nicht-Colorado

Omniscale

Seeding mit Coverages einschränken

Beschränken auf Polygon

```
coverages:
  germany:
    datasource: 'boundary.geojson'
    srs: 'EPSG:4326'
```

Omniscale

Seeding mit Coverages einschränken

Beschränken auf Polygon

```
coverages:
  germany:
    datasource: 'boundary.geojson'
    srs: 'EPSG:4326'
```

Beliebige OGR Datenquellen

```
coverages:
  germany_poly:
    datasource: 'shps/world_boundaries_m.shp'
    where: 'CNTRY_NAME = "Germany"'
    srs: 'EPSG:3857'
```

Omniscale

Wie viele Kacheln?

```
% mapproxy-util grids mapproxy.yaml --seed seed.yaml --coverage germany_poly
utm32_adv:
    Configuration:
        bbox: [-46133.17, 5048875.26857567, 1206211.10142433, 6301219.54]
        min_res: 4891.96981025128
        num_levels: 15
        origin: 'ul'
        srs: 'EPSG:25832'
        tile_size: [256, 256]
    Coverage: germany_poly covers approx. 22.9214% of the grid BBOX
    Levels: Resolutions, # x * y = total tiles (approx. tiles within coverage)
        00:  4891.96981025128,    #      1 * 1      =          1 (0)
        01:  2445.98490512564,    #      2 * 2      =          4 (0)
        02:  1222.99245256282,    #      4 * 4      =         16 (3)
        03:  611.49622628141,     #      8 * 8      =         64 (14)
        04:  305.748113140705,    #     16 * 16     =        256 (58)
        05:  152.8740565703525,   #     32 * 32     =       1024 (234)
        06:  76.43702828517625,   #     64 * 64     =       4096 (938)
        07:  38.21851414258813,   #    128 * 128    =      16384 (3755)
        08:  19.109257071294063,  #    256 * 256    =      65536 (15021)
        09:  9.554628535647032,   #    512 * 512    =     262144 (60087)
        10:  4.777314267823516,   #   1024 * 1024   =      1.05M (240348)
        11:  2.388657133911758,   #   2048 * 2048   =      4.19M (961392)
        12:  1.194328566955879,   #   4096 * 4096   =     16.78M (   3.85M)
        13:  0.5971642834779395,  #   8192 * 8192   =     67.11M (  15.38M)
        14:  0.29858214173896974, #  16384 * 16384  =    268.44M (  61.53M)
```

Omniscale

Nur veränderte Daten aktualisieren

```
coverages:
  changes:
    datasource: "PG: dbname='db' host='host' user='user' password='password'"
    where:
        SELECT geometry
        FROM places
        WHERE last_change >= NOW() - '1 day'::INTERVAL'"
    srs: 'EPSG:25832'
```

Omniscale

Nur veränderte Daten aktualisieren

```
coverages:
  changes:
    datasource: "PG: dbname='db' host='host' user='user' password='password'"
    where:
        SELECT ST_Buffer(ST_Envelope(geometry), 50, 1)
        FROM places
        WHERE last_change >= NOW() - '1 day'::INTERVAL'"
    srs: 'EPSG:25832'
```

Omniscale

Nur veränderte Daten aktualisieren

```
coverages:
  changes:
    datasource: "PG: dbname='db' host='host' user='user' password='password'"
    where:
        SELECT ST_Buffer(ST_Envelope(geometry), 50, 1)
        FROM places
        WHERE last_change >= NOW() - '1 day'::INTERVAL'"
    srs: 'EPSG:25832'
```

Geometrien werden von MapProxy optimiert und Überlappungen entfernt.

Omniscale


Omniscale

Zusammenfassung

- Mobile Anwendungen
- Arbeiten mit bestehenden Kacheln
- Nachträgliche Bildbearbeitung
- Absicherung von Diensten
- Effizientes Seeding

Omniscale

Fin

Danke! Fragen?

Folien:

- https://talks.omniscale.de/2016/fossgis/mapproxy/

MapProxy:

- `mapproxy.org`
- `github.com/mapproxy`
- `mapproxy@lists.osgeo.org`

Me:

- Oliver Tonnhofer, Omniscale
- `tonnhofer@omniscale.de`
- `@oltonn`

Omniscale