

The MapOSMatic Rendering API

Render printable maps without a lot of clicking

Hartmut Holzgraefe

hartmut@php.net

SOTM 2022 - Aug. 19, 2022

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

Who am I?

- Hartmut Holzgraefe
- from Bielefeld, Germany
- studied electric engineering and computer science
- OpenStreetMapper since 2007
- Database Support Engineer for MariaDB Corp.
(and prev. MySQL, Sun, Oracle, SkySQL)



Who am I?

- Hartmut Holzgraefe
- from Bielefeld, Germany
- studied electric engineering and computer science
- OpenStreetMapper since 2007
- Database Support Engineer for MariaDB Corp.
(and prev. MySQL, Sun, Oracle, SkySQL)



Who am I?

- Hartmut Holzgraefe
- from Bielefeld, Germany
- studied electric engineering and computer science
- OpenStreetMapper since 2007
- Database Support Engineer for MariaDB Corp.
(and prev. MySQL, Sun, Oracle, SkySQL)



Who am I?

- Hartmut Holzgraefe
- from Bielefeld, Germany
- studied electric engineering and computer science
- OpenStreetMapper since 2007
- Database Support Engineer for MariaDB Corp.
(and prev. MySQL, Sun, Oracle, SkySQL)



Who am I?

- Hartmut Holzgraefe
- from Bielefeld, Germany
- studied electric engineering and computer science
- OpenStreetMapper since 2007
- Database Support Engineer for MariaDB Corp.
(and prev. MySQL, Sun, Oracle, SkySQL)

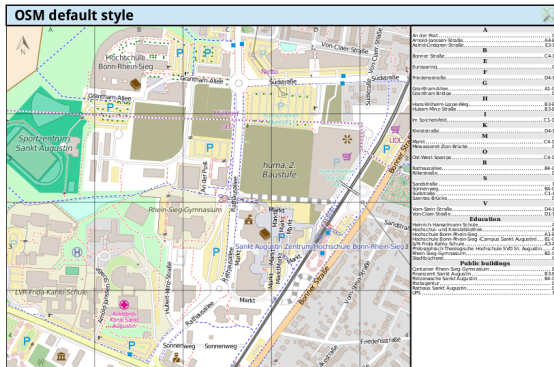


Outline

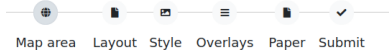
- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

What is MapOSMatic?

A web frontend and backend renderer infrastructure for rendering large format printable maps in various file formats.



Map Area

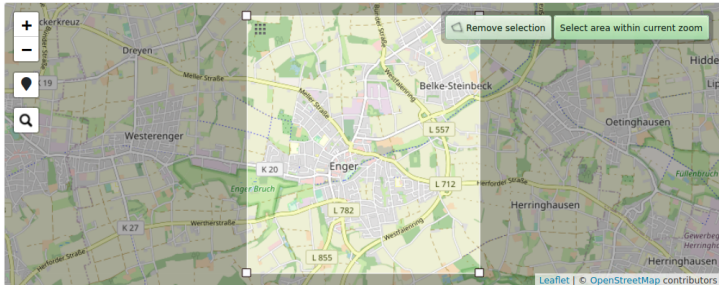


→ Next

Geographic area

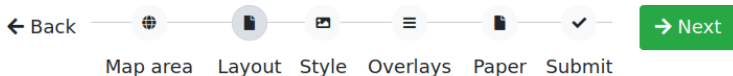
City search

File upload



52°9'32"N 8°32'18"E ↘ 52°7'32"N 8°35'15"E (ca. 3 x 4 km²)

Layout



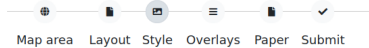
Layout

- ☒ Full-page layout without street index
- ☐ Full-page layout with the street index on the side
- ☐ Full-page layout with the street index at the bottom
- ☐ Full-page layout with the street index on extra page (PDF only)



Map Base Style

← Back



→ Next

Stylesheet

Current CartoCSS OSM style

Current CartoCSS OSM style

Special Interest

The Maposmatic printable stylesheet

HOT Humanitarian style

OpenTopoMap

Current CartoCSS OSM style without street names

OpenOrienteeringMap Blueprint style

Baumkarte by Oliver Rudzick

Black and White

B&W Variant of CartoCSS OSM style

Toner style by Stamen / GeoFabrik

OpenOrienteeringMap Whiteprint style

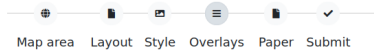
Toner style with roads only



e of the map itself. Note that the stylesheet also drives

Overlay Styles

← Back



→ Next

Overlays

× Compass rose

× Scale bar

Decoration

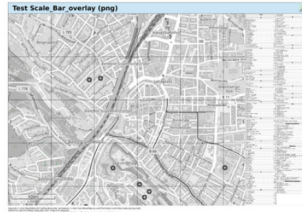
Compass rose

Scale bar

QRcode with request URL

UTM Grid

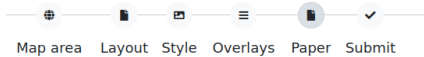
Heights



se style. Multiple overlays can be selected to add
map.

Paper Size

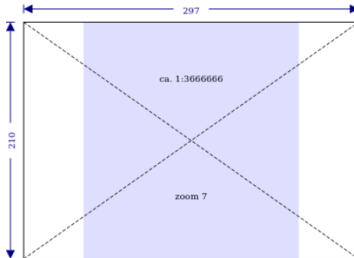
← Back



→ Next

Paper size (width x height)

297 mm ↔ × 210 mm ↓

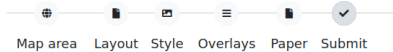


Paper size suggestions

-  **Best fit** (100×110mm²)
-   **Din A4** (210×297mm²)
-   **Din A3** (297×420mm²)
-   **Din A2** (420×594mm²)
-   **Din A1** (594×841mm²)
-   **US letter** (216×279mm²)

Final Step

← Back



Generate ✓

Map title

Locale

 Deutschland (de_DE) ▼

Rendering in Progress

Rendering status

Request submitted

0 minutes ago

Waiting for rendering to begin...

1 / 1

Pending...

✕ Cancel

Updating in 2s...

Finished

Rendering status

Request submitted	5 minutes ago
Rendering started	1 minute ago, after 4 minutes in the queue
Rendering completed	0 minutes ago, after 0 minutes



Rendering was successful.

Recreate ↺

Downloads



PNG
(2.9 MB)



SVGZ
(2.3 MB)



PDF
(2.8 MB)



8BIT.PNG
(1.0 MB)



JPG
(594.6 KB)

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

General API Information

To allow for automated requests without having to click through the user interface a HTTP API has been added with following properties:

- Request parameters (if any) are passed as JSON
- Results are passed as JSON, too
- Most calls are stateless
- Actual render call is returning state information though
- ... to be used in further calls

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - **First Steps**
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

A simple request

Using the curl tool to submit HTTP requests a most basic rendering request may look like this:

```
curl --form job='{ "bbox": [52.0, 8.5, 52.02, 8.52] }'  
https://api.get-map.org/apis/v1/jobs
```

The First Reply

Returning a status reply like this on success:

```
{
  "id": 230035,
  "queue_size": 11,
  "status": 0,
  "status_msg": "Submitted",
  "files": {},
  "interactive": "https://print.get-map.org/maps/230035",

  "language": "en_US.UTF-8",
  "bbox_bottom": 52.02,
  "bbox_left": 8.52,
  "bbox_right": 8.5,
  "bbox_top": 52.0,
  "layout": "plain",
  "paper_height_mm": 297,
  "paper_width_mm": 210,
  "style": "CartoOSM",
  "title": ""
```

The First Reply - Key Parts

The status information enlarged:

```
{  
  "id": 230035,  
  "queue_size": 11,  
  "status": 0,  
  "status_msg": "Submitted",  
  "files": {},  
  "interactive": "https://print.get-map.org/ma  
  ...  
}
```


Checking The Status

The job moves closer to the head of the queue:

```
curl https://api.get-map.org/apis/v1/jobs/230035
```

```
{  
  "id": 230035,  
  "queue_size": 6,  
  "status": 0,  
  "status_msg": "Submitted",  
  "files": {},  
  ...  
}
```

Checking The Status - Again

```
curl https://api.get-map.org/apis/v1/jobs/230035
```

Now the job is getting rendered:

```
{  
  "id": 230035,  
  "status": 1,  
  "status_msg": "In Progress",  
  "files": {},  
  ...  
}
```

Checking The Status - Final

And now the job is done and we can retrieve the results:

```
{  
  "id": 230035,  
  "status": 2,  
  "status_msg": "Done",  
  "files": {  
    "8bit.png": "https://print.get-map.org/results/  
    "jpg": "https://print.get-map.org/results/..."  
    "pdf": "https://print.get-map.org/results/..."  
    "png": "https://print.get-map.org/results/..."  
    "svgz": "https://print.get-map.org/results/...
```

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - First Steps
 - Getting More Complex**
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

A more complex request

```
curl --form job='{  
  "bbox": [52.0, 8.5, 52.02, 8.52],  
  "title": "curl test",  
  "language": "de_DE.UTF-8",  
  "layout": "single_page_index_bottom",  
  "style": "OsmBright",  
  "overlays": ["ContourOverlay", "MaxspeedOverlay"],  
  "paper_size": "Din A1",  
  "orientation": "landscape",  
}' \  
https://api.get-map.org/apis/v1/jobs
```

Page Layouts

<https://api.get-map.org/apis/v1/layouts>

```
{
  "multi_page": {
    "description": "A multi-page layout.",
    "preview_url": "https://api.get-map.org/media/img/layout/multi_page.png"
  },
  "plain": {
    "description": "Full-page layout without index.",
    "preview_url": "https://api.get-map.org/media/img/layout/plain.png"
  },
  "single_page_index_bottom": {
    "description": "Full-page layout with the index at the bottom.",
    "preview_url": "https://api.get-map.org/media/img/layout/single_page_index_..."
  },
  "single_page_index_side": {
    "description": "Full-page layout with the index on the side.",
    "preview_url": "https://api.get-map.org/media/img/layout/single_page_index_..."
  }
}
```

Base Layer Styles

<https://api.get-map.org/apis/v1/styles>

```
{
  "CartoOSM": {
    "annotation": "OpenStreetMap Carto standard style",
    "description": "CartoCSS OSM standard style",
    "preview_url": "https://api.get-map.org/media/img/style/CartoOSM.png"
  },
  "GermanCartoOSM": {
    "annotation": "German OSM style based on OSM Carto",
    "description": "German OSM style",
    "preview_url": "https://api.get-map.org/media/img/style/GermanCartoOSM.png"
  },
  [...]
}
```

Overlay Styles

<https://api.get-map.org/apis/v1/overlays>

```
{
  "OpenRailwayMap_Overlay": {
    "annotation": "OpenRailwayMap overlay",
    "description": "OpenRailwayMap rail line overlay",
    "preview_url": "https://api.get-map.org/media/img/style/OpenRailwayMap_Over
  },
  "Scale_Bar_overlay": {
    "annotation": "",
    "description": "Map scale bar"
    "preview_url": "https://api.get-map.org/media/img/style/Scale_Bar_overlay.j
  },
  [...]
}
```


Paper Formats

https://api.get-map.org/apis/v1/styles/paper_1

```
{  
  "Best fit": {  
    "height": null, "width": null  
  },  
  "Din A4": {  
    "height": 297, "width": 210  
  },  
  "US letter": {  
    "height": 279, "width": 216  
  },  
  "[...]"  
}
```

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - First Steps
 - Getting More Complex
 - Adding Import Files**
 - Example Applications
 - Planned Features
- 4 Wrapping it up

Import File Support

Like the web frontend the API allows to add files that provide additional data to render on top of the base map.

- Supports GPX, general GeoJSON and Umap exports
- Files can be transmitted as direct uploads
- ... or via external URLs
- Bounding box and titles can be determined automatically

GPX Tracks from URL

```
curl --form job='{
    "style": "OsmBright",
    "paper_size": "Din A1",
    "orientation": "portrait",
    "import_urls": [
        "https://get-map.org/example1.gpx",
        "https://get-map.org/example2.gpx"
    ]
}' \
https://api.get-map.org/apis/v1/jobs
```

GPX Tracks from local files

```
curl --form job='{ "paper_size": "Din A1",  
                  "orientation": "portrait"  
--form file1=@example1.gpx \  
--form file2=@example1.gpx \  
https://api.get-map.org/apis/v1/jobs
```

PHP Example

```
<?php
require_once 'HTTP/Request2.php';

define('BASE_URL', 'https://api.get-map.org/apis/v1/');
define('GPX_FILE', 'x.gpx');

$data = ["style" => "OsmBright",
         "paper_size" => "Din_A1",
         "orientation" => "portrait"];

$request = new HTTP_Request2(BASE_URL . "jobs");

$request->setMethod(HTTP_Request2::METHOD_POST)
    ->addPostParameter('job', json_encode($data))
    ->addUpload('track', GPX_FILE);

$reply = json_decode($request->send()->getBody());

echo $reply->interactive."\n";
```

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications**
 - Planned Features
- 4 Wrapping it up

Neighborhood POIs

- Alternative Web Frontend
- Allows for interactive entry of POIs
- Submits actual render request to MapOSMatic
- Forwards user to interactive result page

<https://around.get-map.org/>

City Hiking Atlas

This is a proof-of-concept script for now that:

- Takes OSM id of a city
- Retrieves hiking routes via OverPass API
- Submits render requests for each route
- ... using WayMrakedTrails route GPX URLs
- Waits for all requests to complete
- Stitches results together into one PDF

It may become a full interactive application at a later date ...

<https://>

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API**
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features**
- 4 Wrapping it up

Cancel submitted requests

The web user interface allows to cancel submitted jobs as long as they are still waiting in the queue.
A similar API call is still missing.

Multiple jobs via single request

Less API calls needed when requesting multiple related maps.

Also makes clear that certian jobs are related to each other, and may allow to cancel them all together.

Job prioritization

Change job handling from “first come, first serve” to a more ‘clever’ scheduling scheme.

User authentication

Limit API access to registered users only.
Also combined with job prioritization allows for more fair resource allocation.

Outline

- 1 Introduction
- 2 Quick MapOSMatic Walkthrough
- 3 API
 - First Steps
 - Getting More Complex
 - Adding Import Files
 - Example Applications
 - Planned Features
- 4 Wrapping it up

Wrapping it up

- Try it out and provide feedback :)
- But try to not overload the server
- Consider to run your own instance for more intensive use cases

Questions? Suggestions? Wishes?



References

API documentation

<https://print.get-map.org/about/api/>

My MapOSMatic Instance <https://print.get-map.org/>

GitHub Projects

maposmatic web interface

<https://github.com/hholzgra/maposmatic>

maposmatic render script

<https://github.com/hholzgra/ocitysmap>

maposmatic vagrant VM [https://github.com/hholzgra/
maposmatic-vagrant](https://github.com/hholzgra/maposmatic-vagrant)