Institut für Formale Methoden der Informatik

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Fachstudie Nr. 177

Übersicht und Evaluation am Markt erhältlicher Systeme zur graphischen Darstellung von Kartenmaterial

Jonas Scheurich, Martin Scholz, David Steinhart

Studiengang: Softwaretechnik

Prüfer/in: Prof. Dr. S. Funke

Betreuer/in: Dipl.-Inf. Martin Seybold

Beginn am: 11.04.2013

Beendet am: 11.10.2013

CR-Nummer: A. General Literature,
A.1 INTRODUCTORY AND SURVEY,
H.5 INFORMATION INTERFACES AND PRESENTATION,
H.5.m Miscellaneous

Kurzfassung

Diese Arbeit bietet einen Überblick über bis Anfang Juli 2013 erhältliche Karten-Renderer. Grundsätzlich wird hierbei unterschieden zwischen kommerziellen und gemeinnützigen Projekten. Ein Projekt wird dann als kommerziell bezeichnet, wenn Karten gegen Geld bereitgestellt werden oder eine größere Organisation die Entwicklung der Karten in Auftrag gegeben hat. Gemeinnützige Projekte sind solche, die ihre Karten kostenlos zur Verfügung stellen, ohne finanzielle Absichten zu verfolgen. Es stellt sich heraus, dass viele der nicht-kommerziellen Projekte versuchen neue Techniken einzusetzen, wie zum Beispiel das On-The-Fly-Rendern oder 3D-Darstellung von Kartenmaterial. Im Gegensatz dazu setzen kommerzielle Hersteller weitestgehend auf statische Techniken, zum Beispiel das Rendern von Karten im Vorhinein, welche anschließend nur geladen und angezeigt werden müssen. Des Weiteren setzen die kommerziellen Hersteller auf breite Marktabdeckung. Diese bieten ihre Karten meistens in allen gängigen Formaten an (als Webanwendung, als JavaScript-Plugin, als Plugin bzw. App für iOS und Android), wohingegen die nicht-kommerziellen Projekte meistens nur ein einziges Medium bedienen.

Die Nutzung von vorgerenderten Karten hat den großen Vorteil, dass hochwertigere Karten ohne Verzögerung angezeigt werden können. Zudem muss der Karten-Renderer, der den größten Aufwand erzeugt, nicht auf verschiedene Systeme portiert werden. Vor allem On-The-Fly-Renderer sind oft langsam und haben dennoch qualitativ schlechte Karten, weil auf langsame Optimierungsalgorithmen verzichtet werden muss. Der Vorteil von On-The-Fly-Renderern ist, dass man eine Karte nach eigenen Wünschen verändern kann (z.B. Radwege hervorheben), allerdings gibt es wenige Anwendungen, wo dies von Nutzen wäre.

Diese Arbeit versucht für verschiedene Anwendungsfälle Empfehlungen auszusprechen. Wenn man z.B. eine Anwendung entwickelt, die Karten in veränderter Form benutzen möchte (z.B. andere Optik, personalisierte Karten), so empfiehlt sich die Nutzung eines Freeware-Produkts. Diese sind oft quelloffen und haben geringe Auflagen, allerdings ist die Dokumentation meist mangelhaft und viele der Produkte sind unausgereift. Für die meisten Anwender reicht eines der kostenlosen Pakete eines kommerziellen Anbieters, da diese (in einem festgelegten Rahmen) ohne großen Aufwand genutzt werden können.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Karten sind schon immer ein wichtiges Werkzeuge für den Menschen. Ohne Karten findet man sich in neuen Umgebungen schwer zurecht, und auch an bekannten Orten erleichtern sie die Orientierung. Es ist also kein Zufall, das Karten in vielen Softwareprodukten wie selbstverständlich zur Verfügung stehen. Da Karten mittlerweile nicht nur in Navigationsprogrammen, sondern auch in sozialen Netzwerken, Spielen und vielen anderen Anwendungen genutzt werden, steigt der Bedarf nach einfach nutzbaren Karten weiterhin. Jedoch müssen gleichzeitig Karten nach den vielseitigen Wünschen der Nutzer angepasst werden können.

Zum Einbinden von Karten stehen zur Zeit verschiedene Möglichkeiten zur Verfügung. Der bekannteste Anbieter ist wohl GoogleMaps, welcher schon seit längerer Zeit statische Karten kostenlos zur Verfügung stellt. Im Gegensatz dazu steht OpenStreetMap. Dies ist eine offene Community, welche in Eigenregie Kartendaten sammelt und jedermann zur Verfügung stellt. Es existieren mittlerweile zahlreiche Kartenrenderer, welche OpenStreetMap-Daten nutzen, um Karten zu erzeugen. Die Nutzung dieser Produkte ist meist kostenlos, aber mit größerem Aufwand verbunden. Zudem stellt sich hier oft die Frage, welches das richtige Produkt für welche Aufgabe ist.

Wir haben in dieser Arbeit ca. 30 Dienste verglichen, die verschiedene Arten von Karten zur Verfügung stellen. Wenn man nun selbst eine Karte in ein Projekt einbinden möchte, so muss man sich nicht wie bisher durch ein Meer von Anbietern wühlen, sondern findet in diesem Dokument alle wichtigen Informationen übersichtlich dargestellt.

1.1 Zielsetzung

Diese Fachstudie hatte zum Ziel, die verfügbaren Angebote und Systeme, die Kartenmaterial als Kartenausschnitte anbieten oder auf der eigenen Rechnerarchitektur graphisch darstellen zu vergleichen. Der Vergleich der Produkte sollte mittels Prüfaspekten aus den Bereichen Finanzen, Funktionsumfang des Produktes, Darstellung der Karte, Plattformen und Voraussetzungen, sowie der Installation erfolgen. Abschließend wurden für mehrere Anwendungsfälle Produkte bewertet, die sich für diesen Einsatzzweck eignen. [Fun13]

Zum einfachen Vergleich sollte eine tabellarische Übersicht über alle Produkte erstellt werden.

2 Prüfaspekte

Dieses Kapitel beschreibt die Prüfaspekte. Der Wertebereich '0-5' beschreibt eine subjektive Bewertung. Der Wert 0 stellt die schlechteste, der Wert 5 die beste Bewertung dar.

2.1 Finanzen

Prüfaspekt	Wertebereich	Beschreibung
Lizenz	Text	Angabe zur Lizenzierung.
Open Source	ja/nein	Öffentliche Zugänglichkeit des Source Codes.
Kosten	Preisspanne	Lizenzkosten für den Renderer ohne weitere Hard- oder Softwarekosten.

Tabelle 2.1: Prüfaspekte Finanzen

2.2 Funktionsumfang

Prüfaspekt	Wertebereich	Beschreibung
Interaktive Anzeige	ja/nein	Die Karte kann beim Erstellungsprozess verschoben und gezoomt werden.
Karten offline verarbeiten	ja/nein	Das Kartenmaterial steht auch offline zur Verfügung.
API	ja/nein	Die Funktionalität wird über ein API angeboten.
GUI	ja/nein	Es steht ein GUI zum Einleiten des Rendering-Vorgangs zur Verfügung.
Parametrierung	ja/nein	Die Ansicht kann mit Regeln parametrisiert werden.
Automatischer Kartendownload	ja/nein	Das Kartenmaterial wird automatisch heruntergeladen.
Höhendaten für Route	ja/nein	Die Höhendaten werden ermittelt und angezeigt.
Overlay	ja/nein	Es können externe Daten als Overlay in die Karte eingebunden werden.
Kartenquelle	Text	Quelle des Kartendatenmaterials.
Grafikkartennutzung	ja/nein	Hardwarebeschleunigtes Rendern.
Routenberechnung	ja/nein	Berechnung einer Route.

Tabelle 2.2: Prüfaspekte Funktionsumfang

2.3 Darstellung

Prüfaspekt	Wertebereich	Beschreibung
Subjektiver Eindruck	0-5	Eindruck der Quantität und Qualität der Darstellung des Kartenmaterials.
Performance	0-5	Geschwindigkeit eines Rendervorgangs.
Labeling	ja/nein	Darstellung von Beschriftungen.
Ausgabe	Text	Formate der Renderergebnisse.

Tabelle 2.3: Prüfaspekte Darstellung

2.4 Voraussetzungen

Prüfaspekt	Wertebereich	Beschreibung
Server	ja/nein	Ein (leistungsstarker) Server ist für den Betrieb nötig.
Datenbank	ja/nein	Eine Datenbank ist nötig, um die Kartendaten abzuspeichern.

Tabelle 2.4: Prüfaspekte Voraussetzungen

2.5 Plattformen

Prüfaspekt	Wertebereich	Beschreibung
Plattform	Text	Aufzählung der unterstützten Plattformen.

Tabelle 2.5: Prüfaspekte Plattformen

2.6 Installation

Prüfaspekt	Wertebereich	Beschreibung
Sprache	Text	Implementierungssprachen des Renderers.
Kompiliert	ja/nein/-	Es existiert eine kompilierte Version.
Installer vorhanden	ja/nein/-	Es ist eine automatische Installation vorhanden.
Dauer	Zeit	Dauer der Installation.
Weiterentwicklung	ja/nein/nur Verwaltung	Es findet eine aktive Weiterentwicklung statt, bzw. die Software wird nur noch verwaltet.
Lauffähing	ja/nein	Es existiert eine stabile ausführbare Version der Software.
Einsetzbar	ja/nein	Der Renderer kann in einem Produktiveinsatz genutzt werden.

Tabelle 2.6: Prüfaspekte Installation

3 Kartenquellen

Um Karten grafisch zu erstellen, sind Kartenrohdaten nötig, die von verschiedenen Anbietern zur Verfügung gestellt werden. Die Art und Weise der Veröffentlichung ist sehr unterschiedlich. Teilweise werden die Rohdaten nicht veröffentlicht. Stattdessen sind nur Kartenausschnitte (Tiles) beziehbar.

3.1 AND

Der Kartendaten Anbieter Automotive Navigation Data (AND)[1] bietet Kartenmaterial an, dass auf eigenen Rohdaten basiert.

3.2 Apple

Apple stattet seinen Kartendienst mit eigenen Kartenrohdaten aus, die nicht öffentlich zugänglich sind. Das Rohmaterial selbst stammt unter anderem von TomTom [App13b].

3.3 ESRI

ERSO[2] ist ein Anbieter von vielen Produkten zur Verarbeitung von räumlichen Daten. Dazu gehören auch eigene Kartendaten.

3.4 Google

Google stattet seinen Kartendienst mit eigenem Kartenmaterial aus. Die Rohdaten sind nicht öffentlich zugänglich. Das Rohmaterial stammt unter anderem von INEGI[3]

[1] http://www.and.com/
[2] http://www.esri.de/produkte
[3] http://www.inegi.org.mx/

3.5 Microsoft

Microsoft bietet über seinen Dienst Bing Kartenmaterial an. Die Rohdaten sind nicht öffentlich zugänglich und stammen von Nokia.

3.6 Navteq

Navteq[4] ist ein Kartenrohdaten-Anbieter von Nokia. Navteq bietet Kartenmaterial für 196 Länder. Über 400 Merkmale von Wegpunkten sind in der Navteq Datenbank gelistet. Die Rohdaten werden im GDF[5] vertrieben.

3.7 Nokia

Die von Nokia zur Verfügung gestellten Karten basieren unter anderem auf dem Angebot von Navteq, NavInfo[6] und DigitalGlobe[7] [Nok13d].

3.8 Open Street Map

Open Street Map (OSM) ist ein freies Projekt, dessen Kartenmaterial von der OSM Comunity erfasst und gepflegt wird. Das Rohmaterial ist offen unter der 'Open Data Commons Open Database License' [Ope13a] verfügbar. OSM speichert das Rohmaterial im eigenen XML basierten OSM-Datei-Format[8], oder im platzsparenden PBF[9]

3.9 Public Transport Victoria

Public Transport Victoria[10] ist der Verkehrsverbund Victoria in Australien, der spezielles Kartenmaterial und Verkehrsinformationen zu seiner Gegend anbietet.

[4] http://www.navteq.com/deutsch/
[5] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=54610
[6] http://www.navinfo.com
[7] http://www.digitalglobe.com/
[8] http://wiki.openstreetmap.org/wiki/DE:OSM_XML
[9] http://wiki.openstreetmap.org/wiki/DE:PBF_Format
[10] http://ptv.vic.gov.au/getting-around/maps/

3.10 Tele Atlas

Tele Atlas[11] bot Kartenrohdaten an. 2007 wurde das Unternehmen von TomTom übernommen und verschmolzen.

[11] http://www.teleatlas.at/

4 Bewertung der Kartenrenderer

Dieses Kapitel beschreibt die untersuchten Renderer.

4.1 Kommerziell

4.1.1 ADAC Maps

Allgemeine Übersicht

ADAC Maps ist eine vielseitiges Kartenapplikation, die im Browser sowie auch als App für iPhone, iPad und Android Smartphones zur Verfügung steht [ADA13b].

Allgemein dient die Applikation dem Darstellen von Kartendaten, welche nicht von OSM, sondern von verschiedenen Unternehmen bereitgestellt werden. Innerhalb von Europa, Nordamerika, Kanada, Australien und Neuseeland sind die Kartendaten sehr umfangreich. In anderen Bereichen jedoch, zum Beispiel Südamerika oder China, sind für gewöhnliche nur große Städte und Verbindungsstraßen zwischen diesen eingezeichnet.

Die ADAC Maps werden nur für gewerbliche Zwecke vom ADAC e.V. genutzt und stehen nicht zum Einbinden in eigene Projekte zur Verfügung [ADA13a].

Funktionsumfang und Darstellung

Optisch sind die Karten aufgeräumt und übersichtlich, außerdem funktioniert das Laden und Anzeigen flüssig. Zusätzlich zu Straßen und Gebäuden werden auch Höhendaten dargestellt, zum Beispiel werden die Reliefs von Bergen eingezeichnet.

Das eigentliche Hauptziel der Applikation stellt jedoch die Routenplanung dar. Diese ist sehr gut konfigurierbar und kann verschiedenste aktuelle Informationen, wie Stau-Meldungen und Baustellen, mit einbeziehen. Außerdem können interessante Zusatzinformationen, wie zum Beispiel Campingplätze oder Hotels, gezielt ein- und ausgeblendet werden. Ein Kritikpunkt ist, dass maximal etwa drei Zusatzinformationen (Baustellen, Verkehrsmeldungen, etc.) gleichzeitig angezeigt werden können, da sonst die Karte zu großen Teilen von den eingeblendeten Informationen überdeckt wird.

Was oft bemängelt wird und die größte Schwäche der Smartphone-Applikation darstellt, ist das Fehlen einer Navigationsfunktion, nachdem eine Route berechnet wurde.

Installation und Voraussetzungen

Wenn man die Applikation online nutzen möchte, benötigt man lediglich einen modernen Browser. Auf Smartphones geht die Installation sehr schnell und es werden nur wenig Ressourcen benötigt.

Lizenzen

ADAC Maps gehört zum ADAC e.V. und steht ADAC-Mitgliedern kostenlos zur Verfügung. Die Browser-Applikation kann auch von Nichtmitgliedern benutzt werden, die Handy-Apps ist für Nichtmitgliedern kostenlos, enthält aber nicht alle Funktionen. Diese können für ca. 15 Euro freigeschaltet werden.

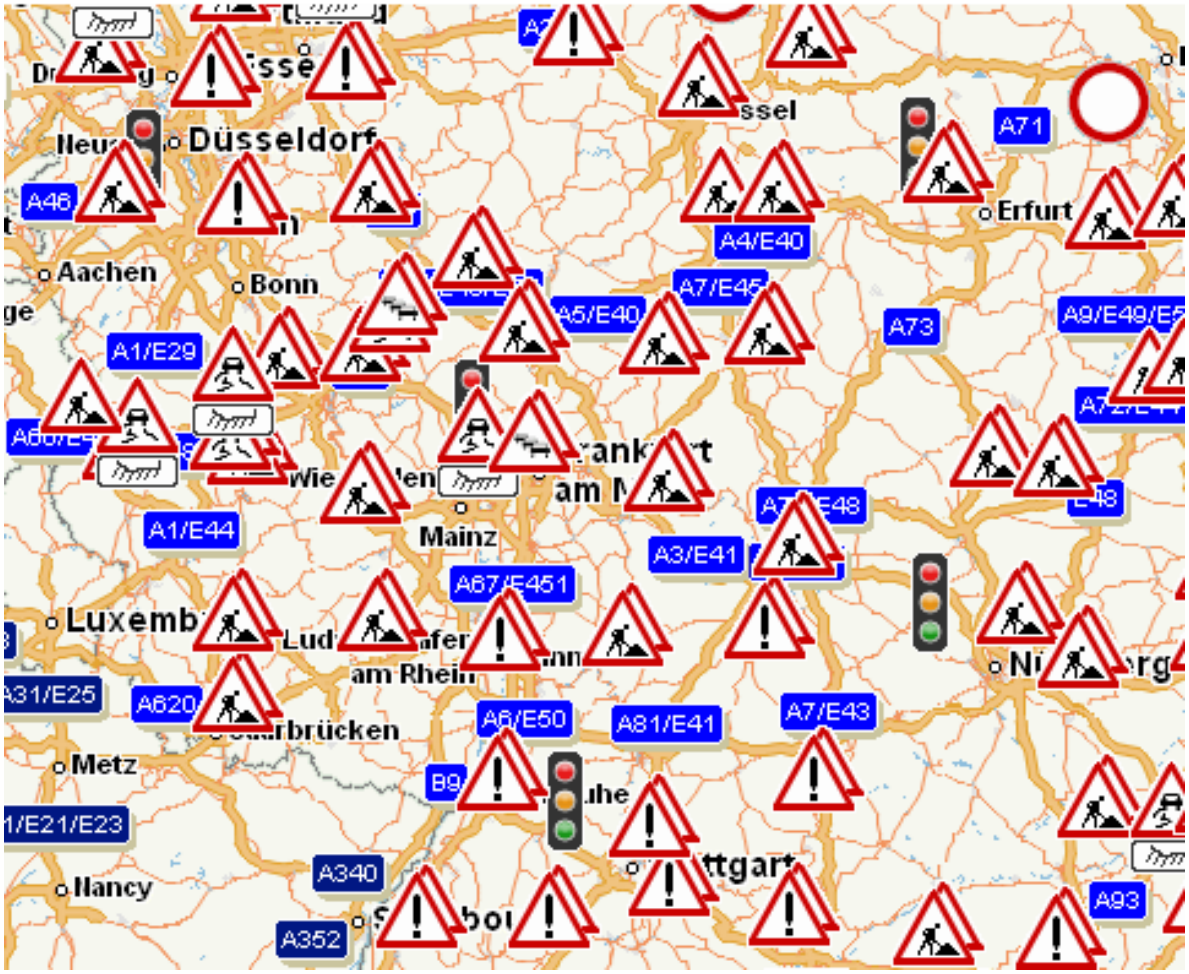

Abbildung 4.1: Deutschland gerendert mit ADAC Maps - wird schnell unübersichtlich

4.1.2 Apple Maps

Allgemeine Übersicht

Apple Maps ist eine Applikation zur Darstellung von Karten auf iOS-Geräten. Die Kartendaten werden online bezogen und stammen von Apple. Es ist also eine ständige Internetverbindung nötig.

Funktionsumfang und Darstellung

Apple Maps bietet viele Funktionen. Neben der 2D-Ansicht steht auch eine 3D-Ansicht zur Verfügung. Dabei werden Satellitenfotos als Textur verwendet, wodurch die Karten sehr realistisch aussehen. Auch die 3D-Modelle der Gebäude sind meist sehr gut. Apple Maps kann auch zur Navigation genutzt werden (vorausgesetzt Apple unterstützt das Gerät und bietet den Service im eigenen Land an), wobei eine Routenführung verwendet wird und Apple Maps somit als vollwertiger Ersatz für ein Navigationsgerät verwendet werden kann. Jedoch muss auch hierzu eine Onlineverbindung bestehen. Die Route kann als Overlay auf der Karte angezeigt werden. Auch die Suche nach POIs wird unterstützt. Gefundene POIs können ebenfalls als Overlay auf der Karte platziert werden. Das Labeling ist sehr übersichtlich. Eine Parametrierung ist nicht möglich - es sind nur die Satellitenbild-Ansicht und die normale Ansicht verfügbar. Insgesamt ist der subjektive Eindruck der Karten sehr gut. Apple bietet ein API an, um Apple Maps in eigene iOS-Apps zu integrieren [App13a].

Installation und Voraussetzungen

Die Installation lässt sich sehr schnell über den AppStore von Apple ausführen. Das iOS-Gerät muss jedoch unterstützt werden. Die Routenführung ist nur in einigen Ländern möglich. Um Apple Maps nutzen zu können, muss eine Internetverbindung bestehen.

Lizenzen

Der Quellcode ist nicht öffentlich zugänglich.

Abbildung 4.2: Stuttgart gerendert mit Apple Maps

4.1.3 Bing Maps

Allgemeine Übersicht

Bing Maps ist eine Applikation, die Karten im Browser anzeigen kann. Es ist auch möglich, Bing Maps in Apps für iOS und Windows Phone zu integrieren [Bin13a]. Die Kartendaten stammen von Nokia und Microsoft.

Funktionsumfang und Darstellung

Die folgenden Beschreibungen beziehen sich auf die Webanwendung von Bing Maps. Bing Maps bietet verschiedene Kartenansichten. Einerseits kann die Karte als normale, gerenderte Karte dargestellt werden, andererseits können Luftbilder als Textur angezeigt werden. Bei den Luftbildern stehen wiederum zwei verschiedene Ansichten zur Verfügung. Es kann eine Perspektive gewählt werden, die die Straßen senkrecht von oben anzeigt. Dabei wird das Satellitenfoto von gerenderten Karten überlagert. Eine zweite Ansicht zeigt die Welt aus einer schrägen Perspektive, sodass auch Fassaden von Häusern sichtbar werden. Bei dieser Ansicht werden keine Overlays angezeigt. Es gibt jedoch keine 3D-Modelle von Gebäuden. Bing Maps bietet einen Routenplanung an. Dabei wird die Route als Overlay auf der Karte angezeigt und es gibt Schritt-für-Schritt-Anweisungen für die Route. Außerdem können aktuelle Verkehrsinformationen in Form von farbig gekennzeichneten Straßen eingeblendet werden. Bing Maps bietet die Möglichkeit, nach POIs zu suchen und diese auf der Karte einzublenden. Auch der eigene Standort kann - falls es der Browser unterstützt - auf der Karte angezeigt werden. Obwohl alle Straßen gelabelt werden, bleibt die Karte übersichtlich. Eine Parametrierung der Karte wird nicht unterstützt. Bing Maps bietet ein API an, um die Karten in eine Webapplikation oder App einzubinden [Bin13a]. Mithilfe der API können auch eigene Overlays in die Karte eingezeichnet werden. Für die Einbindung in Apps ist eine Lizenz nötig, die bis zu einem bestimmten Volumen und unter gewissen Voraussetzungen kostenlos ist [Bin13b].

Installation und Voraussetzungen

Um die Kartenapplikation im Browser ausführen zu können, muss der Browser JavaScript unterstützen.

Lizenzen

Der Quellcode ist nicht öffentlich zugänglich, für die Benutzung der API ist eine Lizenz nötig, die nur bis zu einem bestimmten Volumen kostenlos ist.

Abbildung 4.3: Stuttgart gerendert mit Bing Maps

4.1.4 CartoType

Allgemeine Übersicht

CartoType ist ein API zum Rendern von Karten. Diese steht für eine Vielzahl von Plattformen zur Verfügung. Zu Testen kann eine Demoanwendung von der Herstellerhomepage bezogen werden [Car13e]. Hervorzuheben ist die integrierte Routenberechnung.

Funktionsumfang und Darstellung

Der Kartenrenderer erstellt optisch ansprechende Karten. Das Beispiel vom Bärensee in Stuttgart bringt dies leider nicht gut zur Geltung. Die Kartendaten müssen manuell heruntergeladen werden und werden offline verarbeitet. Um das Kartenmaterial mit CartoType nutzen zu können, muss dieses in das hauseigene Format CTM1 konvertiert werden [Car13d].

Die Karten können mittels Stylesheets an ein bestehendes Layout angepasst werden.

Des weiteren können die Karten mit eigenen Daten oder Routen überlagert werden um kundenspezifische Informationen anzuzeigen. Die Höhendaten werden per Height Coloring, Height Shading oder in Kombination der beiden Techniken visualisiert. Eine 3D Darstellung ist nicht möglich.

Die integrierte Routenberechnung zeigt die Route aus der Vogelperspektive als Overlay an. Es ist möglich eine Route anzuzeigen, sowie Turn-by-Turn-Navigation vorzunehmen. Die Karte kann zur Routenanzeige auch von einem geringeren Winkel als die orthogonale Draufsicht betrachtet werden [Car13a].

Aktuell wird CartoType intensiv weiterentwickelt und hat mit 'Lonely Planet' einen professionellen Kunden.

Installation und Voraussetzungen

Angeboten wird eine C++ API, sowie eine .NET API für die Sprachen C#, Visual Basic .NET, and C++/CLI. (Windows XP, 7, 8) und eine Android Java API. Für Entwicklungen auf den Plattformen iPhone, iPad und Mac OS X, sowie Linux kann die C++ API genutzt werden [Car13b].

Die Dokumentation der API sowie die Einbindung in die jeweiligen Entwicklungsumgebung sind auf der Homepage von CartoType ausführlich beschrieben.

Die Karten werden von OpenStreetMap, Ordnance Survey OpenData[1] und USGS digital elevation data[2] bezogen.

[1]Ordnance Survey OpenData: `http://www.ordnancesurvey.co.uk/`
[2]USGS digital elevation: `http://dds.cr.usgs.gov/srtm/`

Eine Datenbank oder ein Server wird nicht benötigt.

Lizenzen

Die Lizenzkosten von CartoType liegen bei £2,500 für 1000 ausgelieferte Endnutzerlizenzen der mit CartoType entwickelten Software durch den Lizenznehmer. Für die Routenberechnung werden 25 % der Lizenzkosten berechnet. Die Supportkosten belaufen sich auf 25% der Vorjahres-Lizenzkosten. Der Source Code wird mit ausgeliefert [Car13c].

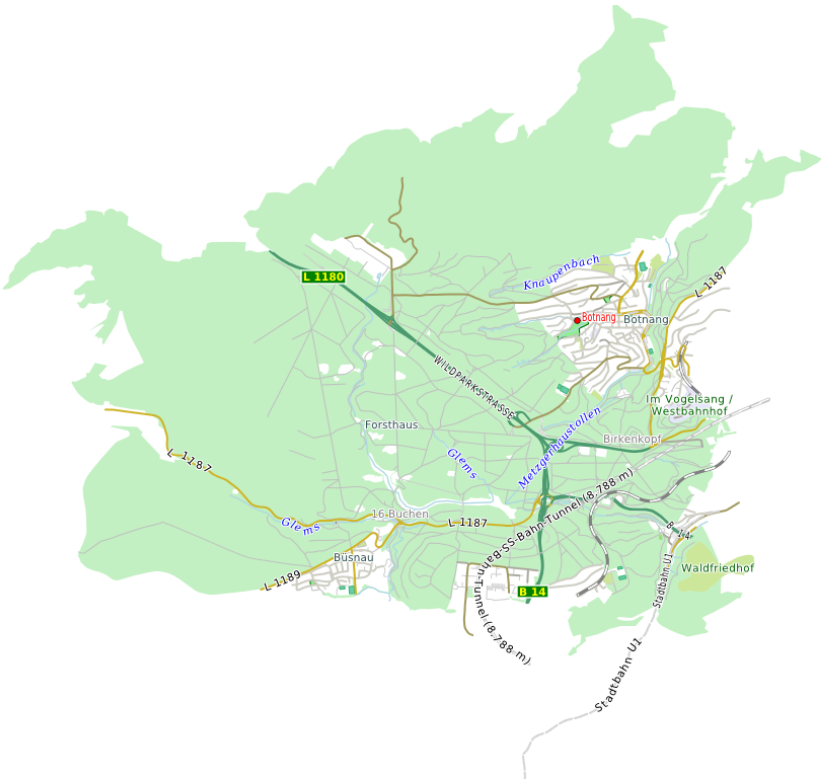

Abbildung 4.4: Stuttgart Bärensee gerendert mit CartoType. Seen werden in der Standartkonfiguration nicht dargestellt.

4.1.5 Google Maps

Diese Erläuterungen zu Google Maps berücksichtigen nicht die Änderungen, die Google ab Juni 2013 schrittweise einführt.

Allgemeine Übersicht

Google Maps wird im Browser, als Android und iPhone App, sowie über eine Reihe von APIs angeboten.

Funktionsumfang und Darstellung

In der Browser-Version ist kein Kartenexport in ein Grafikformat möglich. Allerdings ist es möglich Karten zu bearbeiten und zu personalisieren. Die erstellte Karte kann per Link versendet werden, oder in eine Homepage eingebettet werden. Per Google Maps Labs können weitere Funktionen hinzugefügt werden, wie zum Beispiel das Anzeigen von Längen- und Breitengraden.

Google
Nahverkehr, Fahrradfahrer und Fußgänger an.

Die *Google Maps JavaScript API* [Goo13h] in der Version 3 bietet die Möglichkeit, interaktive Karten in eine Website einzubetten. Die Karten können mit neuen Steuerelementen versehen werden um weitere Funktionen hinzuzufügen. Der Style der eingebetteten Karte kann ebenfalls geändert werden. Um die Karte mit weiteren Informationen zu versehen, können die Karten mit Overlays versehen werden. Neben einer Basis-Klasse für eigene Overlays können Symbole, Pfade, Polylinien, Polygone, Formen und interaktive Zeichnungen des Endnutzers auf der Karte dargestellt werden. Die von Google bereitgestellten Ebenen, wie Verkehrsinformationen, Wetter, Fahrradwege, Verkehrsverbünde, Heatmaps mit eigenen Daten, oder KML- und GeoRSS-Ebenen können einzeln eingeblendet werden..

Die *Google Maps Image API* [Goo13g] ermöglicht es per HTML Karten oder StreetView-Bilder in eigene Homepages einzubinden.

Die *Google Directions API* [Goo13a] ist Teil der Google Maps Webdienste, die eine Sammlung von HTML-Webdiensten sind. Alle diese Dienste stehen auch in der JavaScript API zur Verfügung. Die Ergebnisse können als JSON oder XML zurückgegeben werden. Neben den Standardparametern (Start, Ziel, Standort von einem Sensor erfasst) können weitere optionale Parameter zur Routenberechnung verwendet werden: Transportart, Wegpunkte, Routenalternativen im Ergebnis, Einschränkung der Straßen, Region, Abfahrts- und Ankunftszeiten.

Die *Google Distance Matrix API* [Goo13b] ist Teil der Google Maps Webdienste und berechnet die Fahrtzeiten zwischen zwei Punkten. Die Ergebnisse dürfen nur in Verbindung einer Google Maps Karte verwendet werden.

Die *Google Elevation API* [Goo13c] ist Teil der Google Maps Webdienste und ermittelt Höhendaten an einer Stelle oder entlang eines Pfades.

Die *Google Geocoding API* [Goo13e] ist Teil der Google Maps Webdienste und konvertiert Adressen im String-Format (z.B. Informatik Uni-Stuttgart, Universitätsstraße, Stuttgart, Deutschland) in Längen- und Breitengrade. Diese können zum Beispiel genutzt werden, um Marker auf der Karte zu setzen.

Die *Google Places API* [Goo13j] ist Teil der Google Maps Webdienste und ermöglicht die Suche nach Umgebungsinformationen, Detailinformationen zu einem bestimmten Punkt und Bildern einer Position. Des weiteren wird eine Autovervollständigung für Texteingaben angeboten.

Die *Google Maps API für Unternehmen* [Goo13d] ist die kommerzielle Version der bestehenden Google Maps APIs, die eine höhere Kapazität, Intranet-Support und Kontrolle über die angezeigte Werbung bietet.

Die *Google Earth API* [Goo13f] ermöglicht die Einbindung eines 3D-Modells der Erde.

Die oben beschriebenen APIs können für die mobile Anwendung genutzt werden. Des weiteren steht die *Google Maps Android API* zur Verfügung. Diese bietet ein Teil der Funktionen der Google Maps API Familie. Allerdings steht diese API nicht auf allen Android-Geräten zur Verfügung, da sie nicht Teil des Android SDK ist.

Lizenzen

Die Lizenzierung ist über *Google Maps/Google Earth APIs Terms of Service* geregelt. Google bietet sein API in zwei verschiedenem Paketen an [Goo13i].

Google Maps API - kostenlos
Um dieses Paket nutzen zu können muss die Anwendung kostenlos und öffentlich zugänglich sein.

Geocoding	2.500 Zugriffe/Tag
Routenplanung	2500 Zugriffe/Tag mit je 10 Wegpunkten
Distance Matrix	10 Elemente pro Anfrage, 100 Elemente in 10 Sekunden, insgesamt 2.500 Elemente/Tag
Höhendaten	2.500 Zugriffe/Tag mit 25.000 Stichproben/Tag
Static Maps API	Auflösung: 640 x 640 Skalierung: 2x
Street View API	Auflösung: 640 x 640

Tabelle 4.1: Google Maps API - kostenlos

Google Maps API für Unternehmen - kostenpflichtig
Für folgende Einsatzzwecke muss die kostenpflichtige API genutzt werden:

- Interne Bereitstellung
- Kostenpflichtige Software
- Wiederverkaufsdienste
- Kontrolle von Webanzeigen
- Private Güterverfolgung

Geocoding	100.00 Zugriffe/Tag
Routenplanung	100.000 Zugriffe/Tag mit je 23 Wegpunkten
Distance Matrix	625 Elemente pro Anfrage, 1.000 Elemente in 10 Sekunden, insgesamt 100.00 Elemente/Tag
Höhendaten	100.000 Zugriffe/Tag mit 1.000.000 Stichproben/Tag
Static Maps API	Auflösung: 2048 X 2048 Skalierung: 4x
Street View API	Auflösung: 2048 X 2048

Tabelle 4.2: Google Maps API für Unternehmen - kostenpflichtig

Über den *Google Enterprise Sales Manager* kann ein Angebot über eine OEM Lizenzierung eingeholt werden.

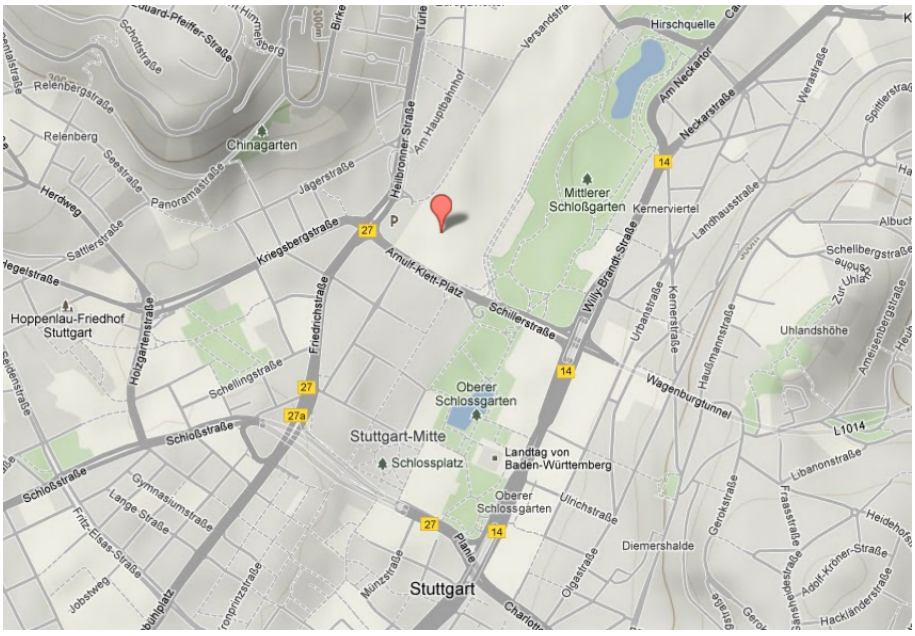

Abbildung 4.5: Stuttgart Hbf gerendert mit Google Maps (Version vor dem Update im Juli 2013). Diese Konfiguration stellt die Höhendaten durch Hill-Shading dar.

4.1.6 Mappy

Allgemeine Übersicht

Mappy ist eine Kartenapplikation für Browser und Smartphones, welche von Mappy SA, einer Tochtergesellschaft der französischen Gesellschaft Solocal (ehemals Pages Jaunes Groupe), bereitgestellt wird [Sol13a]. Mappy kann in eigene Projekte eingebunden werden, wenn diese der Öffentlichkeit kostenlos zugänglich gemacht werden und nicht der Navigation dienen. Hierfür muss mithilfe eines Online-Formulars ein Antrag gestellt werden, welcher von Mappy SA bearbeitet wird. Die Website richtet sich eher an französischsprachige Entwickler, so stehen zum Beispiel viele Erklärungen nur auf Französisch und gelegentlich auf Englisch bereit.

Funktionsumfang und Darstellung

Mappy bietet keine herausragenden Funktionen, verglichen mit anderen Anbietern. Es können Kartentiles angezeigt werden, welche von einem Server bereitgestellt werden. Zudem können Satellitenbilder der Erde angezeigt werden [Sol13c]. Da es sich um eine französische Firma handelt, sind diese Bilder in Frankreich sehr hochauflösend, für den Rest der Welt gibt es jedoch nur Bilder in geringer Auflösung. Das Anzeigen der Karten läuft flüssig, allerdings wirkt die Karte manchmal etwas unübersichtlich [Sol13b].

Installation und Voraussetzungen

Wenn Mappy im Browser benutzt wird, benötigt es keine gesonderte Installation, es reicht ein moderner Browser mit JavaScript. Die Einbindung in eine Smartphone-Applikation kann mithilfe der zahlreichen Beispielimplementierungen relativ einfach vorgenommen werden. Das Einrichten und Warten eines Servers entfällt, da Mappy SA ausschließlich eigene Kartendaten verwendet, welche von verschiedenen Firmen (AND, GeoSignal, TeleAtlas) zusammen getragen wurden, und diese selbst bereitstellt.

Lizenzen

Grundsätzlich ist die Nutzung von Mappy kostenlos, solange das Programm, in das Mappy eingebunden ist, ebenfalls kostenlos ist. Es gibt keine expliziten Angaben, ob die kommerzielle Nutzung gänzlich ausgeschlossen ist. Der Sourcecode steht nicht offen zur Verfügung.

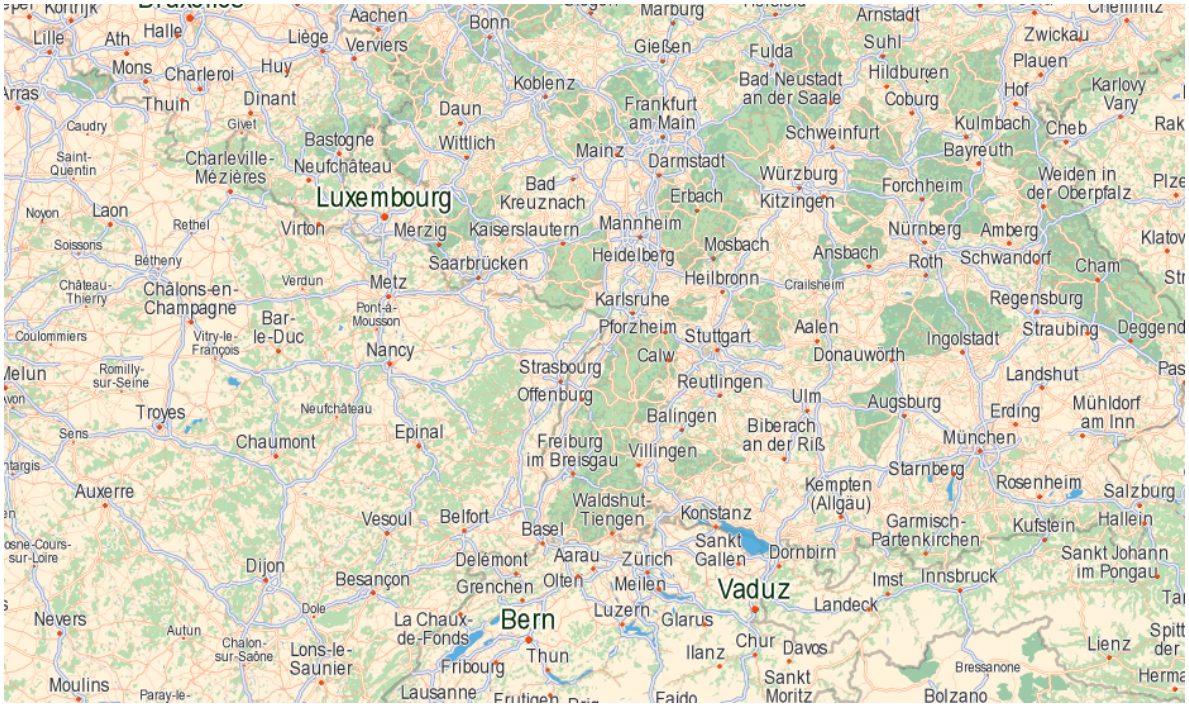

Abbildung 4.6: Deutschland gerendert mit Mappy - Städtenamen sind oft zu klein

4.1.7 MapsWithMe

Allgemeine Übersicht

MapsWithMe ist eine App, die sowohl unter iOS, als auch unter Android läuft. Das besondere an MapsWithMe ist, dass es in der Lage ist, offline zu rendern [Map13n]. Dazu müssen die Kartendaten zuvor heruntergeladen werden. Die Funktion des Offline-Renderns auf Smartphones ist eine große Ausnahme - dies lässt MapsWithMe besonders herausstechen.

Funktionsumfang und Darstellung

MapsWithMe kann Karten von OpenStreetMap offline rendern. Die dazu benötigten Daten werden direkt aus der App heraus heruntergeladen und auf dem Smartphone gespeichert. Dadurch ist es möglich, sich die Karten auch ohne Internetverbindung anzeigen zu lassen. Die Kartendaten sind in einzelne Pakete unterteilt, sodass es möglich ist, nur die Karten für ein Land, oder eine Region zu laden. Trotzdem gibt es immer eine Weltansicht, um sich orientieren zu können.
Die App kann mit den üblichen Gesten gesteuert werden und läuft stets sehr flüssig. Bei den unterstützten Funktionen muss zwischen der kostenlosen und der Pro-Version unterschieden werden. Die kostenlose Version unterstützt ausschließlich das Anzeigen der eigenen Position mittels GPS. Die Pro-Version unterstützt zusätzlich das Suchen nach Adressen und Orten. Auch das Exportieren von Bookmarks ist in dieser Version möglich. Beide Versionen unterstützen Labeling und können POIs als Overlay anzeigen. Eine Routenberechnung ist in keinen der beiden Versionen möglich. Die Pro-Version kostet 3,99 Euro in Google Play [Map130].

MapsWithMe bietet eine API, um die Karte in eigene Anwendungen zu integrieren. Die API steht für iOS und Android zur Verfügung. Sie ermöglicht es, eigene Punkte auf der Karte einzuzeichnen, die Karte mit einem Firmenlogo zu überlagern, sowie die Karte als Auswahl für einen Ort zu benutzen [Map13m].

Installation und Voraussetzungen

Die Installation der App geht sehr einfach über den Google Play bzw. AppStore von Apple. Es muss mindestens Android 2.1 bzw. iOS 5 installiert sein. Die Einbindung von MapsWithMe in eigene Maps sollte relativ einfach sein, da Beispielimplementierungen beiliegen.

Lizenzen

MapsWithMe benutzt eine eigene Lizenz. Der Quellcode ist nicht öffentlich zugänglich. Die Benutzung der API ist jedoch kostenfrei.

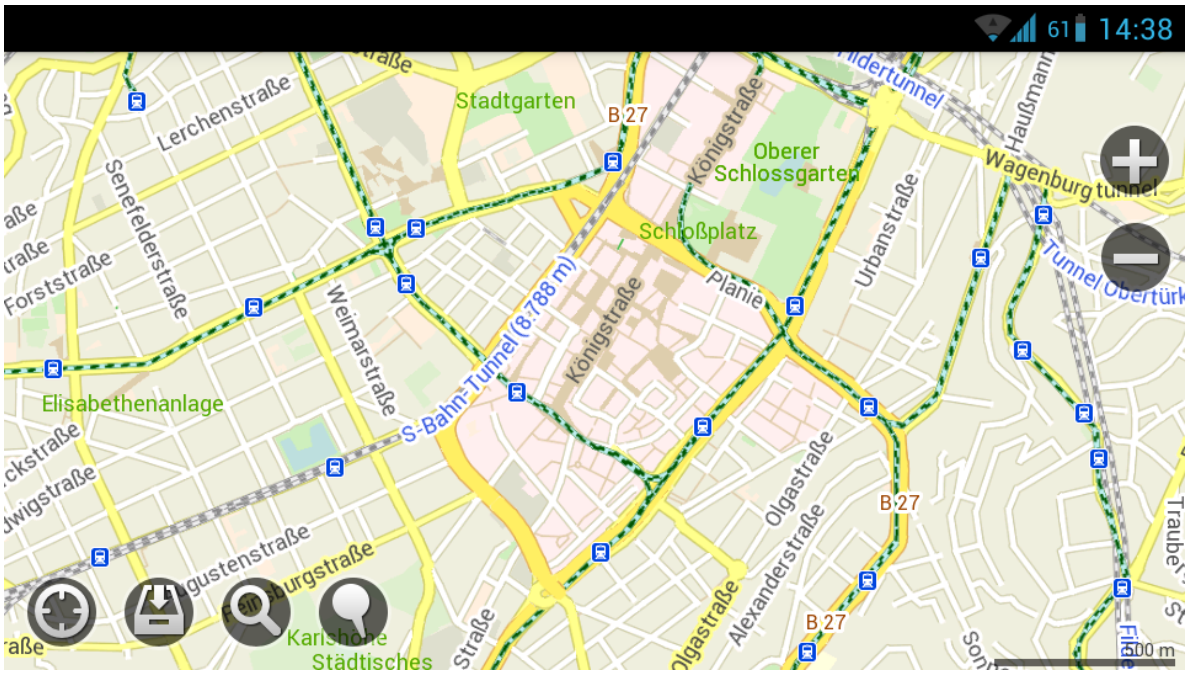

Abbildung 4.7: Stuttgart gerendert mit MapsWithMe

4.1.8 Nokia Maps

Allgemeine Übersicht

Die vom Handyhersteller Nokia vertriebenen Karten werden unter der Marke 'HERE' vertrieben und stehen für eine Vielzahl - meist mobiler Plattformen - zur Verfügung.

HERE bietet unter anderem Routenplanung, personalisierte Karten, sowie 3D Karten mit Satellitenaufnahmen.

Funktionsumfang und Darstellung

Die Karten von HERE sind vorgerendert und können mit Overlays versehen werden. Es stehen verschiedene Ansichten, wie orthogonale Draufsicht, Schrägansicht, Nachtmodus, Satellitenaufnahmen und wenn verfügbar auch 3D Modelle von Gebäuden zur Verfügung. Höhendaten werden neben der 3D-Ansicht auch per Hill-Shading angezeigt. Sämtliche Karten können mit Overlays oder POIs versehen werden. Ein Export in Grafikformate ist nicht möglich.

Die Karten für Javascript-Anwendungen können online in einem WYSIWYG-Editor erstellt werden. Dabei werden einzelne thematischer Layer in die Karte gezogen. Auch können persönliche Overlays eingefügt werden. Der Javascript-Code wird automatisch generiert.

HERE ermöglicht Routenberechnung und Turn-by-Turn-Navigation für Kraftfahrzeuge, Fußgänger und den Öffentlichen Nahverkehr.

In Zukunft sind unter anderem Augmented Reality-Anwendungen geplant.

Installation und Voraussetzungen

HERE wird als Hompage und als App für iOS, Android, Windows Phone 7 und Windows Phone 8 angeboten.

Zur Einbindung in eigene Projekte stehen für Webanwendungen APIs für Javascript, REST und HTML5 zur Verfügung [Nok13b]. Mobile Anwendungen können mit APIs für Windows Phone, Andorid, Qt und Java ME entwickelt werden [Nok13c].

Lizenzen

Nokia vertreibt die HERE Maps in zwei Paketen. Die Abfrage von Kartenmaterial ist grundsätzlich kostenlos [Nok13a].

Base - kostenlos

Karten	• 2D Karten (Normal, Mobil, Fußgänger und Nachtmodus • Comunity Karten • Statische 2D Karten	unbegrenzte Zugriffe
Routenberechnung	• Autorouten • Fußgängerrouten	2.500 Zugriffe/Tag
Suche	• Umgebungssuche • Ortsinformationen	2.500 Zugriffe/Tag

Tabelle 4.3: Here Maps - Base

Core - $1.500 pro Monat

Karten	• 2D Karten (Normal, Mobil, Fußgänger und Nachtmodus • Comunity Karten • Statische 2D Karten	unbegrenzte Zugriffe
Routenberechnung	• Autorouten • Fußgängerrouten • ÖPVN	10.000 Zugriffe/Tag
Suche	• Umgebungssuche • Ortsinformationen • Geocoding • Reverse Geocoding	10.000 Zugriffe/Tag
Verkehr	• Verkehrsinformationen	10.000 Zugriffe/Tag

Tabelle 4.4: Here Maps - Core

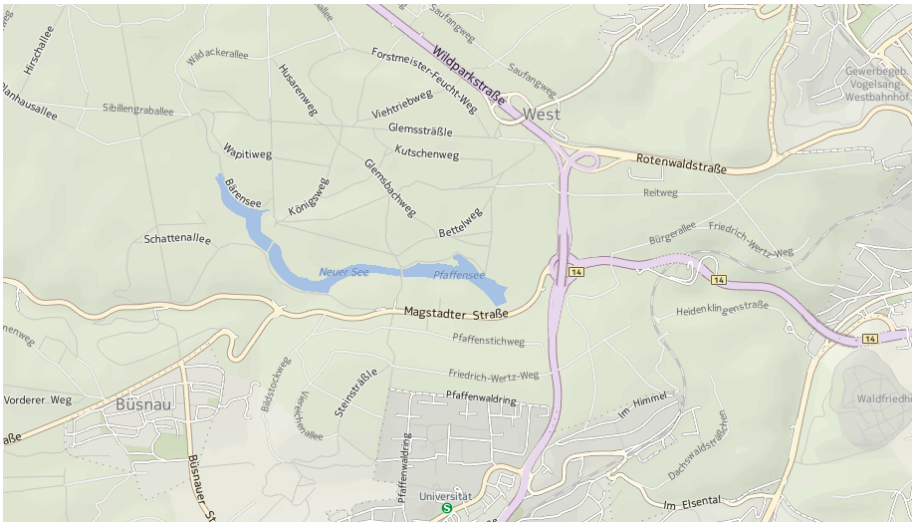

Abbildung 4.8: Stuttgart Bärensee gerendert mit HERE

4.1.9 TileMill

Allgemeine Übersicht

Der Kartenrenderer TileMill nutzt die Render-Engine Mapnik um mittels Stylesheets Karten inklusive eigener Informationsdaten zu erstellen.

Funktionsumfang und Darstellung

Der Kartenrenderer bietet eine graphische Oberfläche um die Änderungen an der Karte sofort anzuzeigen. Die Karten werden mittles CartoCSS Stylesheets angepasst. Weitere Informationen, wie zum Beispiel Straßen, Gebäude, oder eigene Daten werden mit Layern eingebunden. Somit können auch Höhendaten eingebunden werden [Map13d].

Durch die Speicherung der Kartendaten auf dem eigenen Rechner ist ein problemloses Arbeiten ohne dauerhafte Internetverbindung möglich.

Für mobile Anwendungen steht ein SDK für Java Script, iOS und eine REST API [Map13d] zur Verfügung.

Aktuell werden TileMill sowie Mapnik weiterentwickelt.

Die erstellte Karte ist komplett von den eingefügten Layern abhängig. Für jede Information (z.B Länder, Straßen, Gebäude usw.) muss ein zusätzliches Layer eingebunden werden. Über CartoCSS Stylesheets lässt sich die optische Darstellung der Layer an die gegebenen Anforderungen anpassen.

Das Labeling ist kollisionsfrei. Label, die in der eingestellten Zoomstufe nachrangig sind, werden nicht dargestellt.

Die Karten können in folgende Formate exportiert werden:

- PNG
- PDF
- SVG
- MBTiles
- Mapnik XML

Installation und Voraussetzungen

TileMill ist für die Betriebssysteme ab Windows XP, ab Mac OS X und Linux verfügbar. Die Installation ist mittels eines automatischen Installers (Windows) realisiert. Der Hersteller fordert 2GB RAM, sowie eine Internetverbindung [Map13a].

Als Kartenquelle werden OSM Karten von Mapbox verwendet. Diese können aus einzelnen Dateien (die automatisch heruntergeladen werden) aus einer SQLite-, oder einer PostGIS Datenbank bezogen werden.

Das Hosten eines eigenen Kartenservers ist nicht möglich.

Lizenzen

Die Anwendung kann kostenlos heruntergeladen werden. Zur Verwendung ist ein Mapbox Account nötig. Der Source Code ist nicht öffentlich zugänglich. Die Nutzung von TileMill muss direkt von Mapbox lizenziert werden [Map13c].

Für die Nutzung fallen folgende Kosten an [Map13b]:

kostenloses Paket

Preis	kostenlos
Kartenaufrufe/Monat	3.000 Aufrufe

Tabelle 4.5: Mapbox - kostenloses Paket

Basic Paket

Preis	$5/Monat
Kartenaufrufe/Monat	10.000 Aufrufe
Online Support	ja
Satelliten Bilder	ja
Werbefrei	nein (Mapbox Werbung)

Tabelle 4.6: Mapbox - Basic Paket

Standard Paket

Preis	$49/Monat
Kartenaufrufe/Monat	100.000 Aufrufe
Online Support	ja
Satelliten Bilder	ja
Werbefrei	ja

Tabelle 4.7: Mapbox - Standard Paket

Plus Paket

Preis	$149/Monat
Kartenaufrufe/Monat	300.000 Aufrufe
Online Support	ja
Satelliten Bilder	ja
Werbefrei	ja
SSL Aufrufe	ja

Tabelle 4.8: Mapbox - Plus Paket

Premium Paket

Preis	$499/Monat
Kartenaufrufe/Monat	1.000.000 Aufrufe
Online Support	ja
Satelliten Bilder	ja
Werbefrei	ja
SSL Aufrufe	ja
24h Support	ja

Tabelle 4.9: Mapbox - Premium Paket

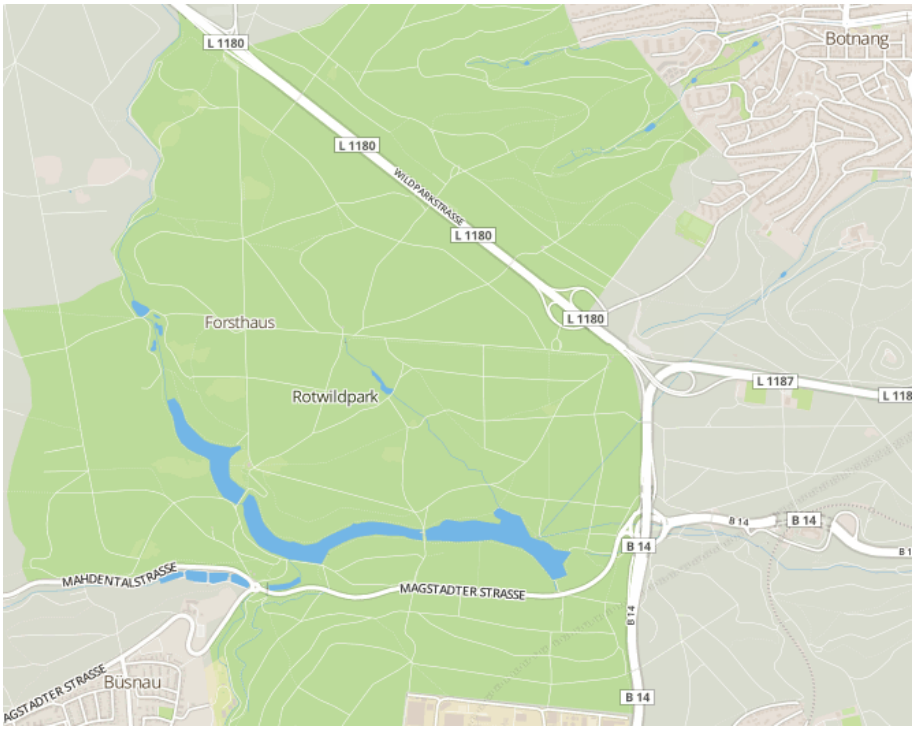

Abbildung 4.9: Stuttgart Bärensee gerendert mit TileMill

4.1.10 TomTom

Allgemeine Übersicht

TomTom ist ein bekannter Hersteller von Navigationsgeräten und hat eine vielfältiges Angebot [Tom13b]. Es gibt Karten für mobile Navigationsgeräte, für Android-Geräte, für iPhone und iPad und für Webbrowser. Zudem gibt es viele Zusatzfunktionen, von der Routenplanung und Navigation bis hin zur Stau-Überwachung. Allerdings stehen außer der Online-Karte diese Angebote nur kostenpflichtig zur Verfügung. TomTom besitzt eigene Kartendaten von 112 Ländern, es gibt aber keine genauen Angaben, wie vollständig diese sind. Es ist möglich, Karten auch offline zu nutzen, was bei Mobiltelefonen zur Zeit eher unüblich ist. Es gibt für Firmen die Möglichkeit, TomTom-Karten in eigene Produkte einzubinden. Man kann zum Beispiel auch nur eine Routenplanung oder eine genaueren Ortsbestimmung in Zusammenhang mit der Kartendarstellung einbinden.

Funktionsumfang und Darstellung

Die TomTom Karten sind optisch hübsch und relativ übersichtlich. Man hat auf das Einblenden von Satellitenbildern verzichtet, da die Karten hauptsächlich der Navigation dienen und Straßen etc. gut zu erkennen sein müssen. Größere Städte werden etwas hervorgehoben und auch Ländergrenzen etc. werden korrekt eingezeichnet. Da sämtliche Karten von TomTom bereit gestellt werden, sind diese nicht parametrierbar [Tom13c].

Lizenzen

TomTom bietet Firmen verschiedene Möglichkeiten, ihre Karten kommerziell einzusetzen. Es gibt keine genauen Angaben, was entsprechende Lösungen kosten und für welche Plattformen diese bereitstehen. Bei Interesse muss man sich direkt an den Support von TomTom wenden [Tom13a].

Abbildung 4.10: Stuttgart gerendert mit TomTom - Ohne Einblendung von Verkehrsinformationen

4.2 Freeware, lauffähig

4.2.1 Ceyx

Allgemeine Übersicht

Der Kartenrender Ceyx ist ein kostenloses Tool, das aktuell nur noch sporadisch weiterentwickelt wird und für Windows, Linux und Mac OS zur Verfügung steht.

Funktionsumfang und Darstellung

Ceyx ist ein Konsolenprogramm, das mittels MapCSS 0.2 parametriert wird. Die zuvor manuell heruntergeladenen Kartendaten im OSM Format werden als 2D PNG-Grafik gerendert. Ein automatischer Kartendatendownload findet nicht statt. Eine Offline-Verarbeitung der Karten ist somit möglich. Die Einbindung von Höhendaten, Overlays oder die Berechnung von Routen ist nicht möglich. Die Beschriftung ist mit einem 6-Punkt Labeling realisiert.

Installation und Voraussetzungen

Der Source Code wird aktuell als instabil bezeichnet [Ope13f]. Ceyx benötigt Python-Cairo, Python-Simplejson und Pangocairo. Für Ceyx liegt kein Installationsscript, oder Installer vor. Es müssen alle Abhängigkeiten selbst installiert und der Renderer manuell kompiliert werden.

Lizenzen

Der Programmcode ist öffentlich unter einer GPL v2+ Lizenz verfügbar [Ope13e].

4.2.2 Kendzi3D

Allgemeine Übersicht

Kendzi3D ist ein Plugin für JOSM, das Karten dreidimensional darstellen kann. Es nutzt dazu Daten, die JOSM dem Plugin zur Verfügung stellt. Da Kendzi3D in Java läuft, sollte es auf jedem Betriebssystem laufen. Kendzi3D ist jedoch noch in einer sehr frühen Entwicklungsphase.

Funktionsumfang und Darstellung

JOSM kann dem Plugin entweder durch Öffnen einer OSM-Datei oder durch das Herunterladen eines ausgewählten Kartenausschnitts die Kartendaten zur Verfügung stellen. Kendzi3D benutzt Standardtexturen und Modelle, um Gebäude generisch anzeigen zu können. Erst, wenn den Gebäuden Tags zugeordnet werden, können Gebäude individualisiert dargestellt werden. Kendzi3D unterstützt die Darstellung verschiedener Dachformen. Es können die Art des Dachs, die Ausrichtung und die Textur angegeben werden [Ken13a]. Auch die Fassaden können individuell für jedes Gebäude gestaltet werden. Es ist möglich, sowohl Fenster, als auch Eingänge durch entsprechende Tags darstellen zu lassen [Ken13b]. Durch die Einbindung eigener 3D Modelle können Objekte wie Briefkästen, Hydranten oder Windkrafträder ebenfalls individualisiert werden. Einige wenige Modelle sind bereits in Kendzi3D vorhanden. Außer der einheitlichen Standardtextur für den Boden ist es möglich, eine vorgerenderte Karte als Bodentextur zu verwenden. Straßen werden jedoch weiterhin über die Bodentextur gezeichnet. Labeling oder Overlays werden nicht unterstützt. Die Steuerung erfolgt mit Maus und Tastatur. Es kann sowohl die Kameraposition, als auch die Blickrichtung frei gewählt werden.

Installation und Voraussetzungen

Die Installation ist sehr simpel. Kendzi3D kann Plugin als in JOSM ausgewählt und direkt durch JOSM heruntergeladen werden. Es ist dann sofort einsetzbar.

Lizenzen

Wir konnten keine Angaben zu Kosten oder der verwendeten Lizenz finden. Der Quellcode ist jedoch öffentlich zugänglich.

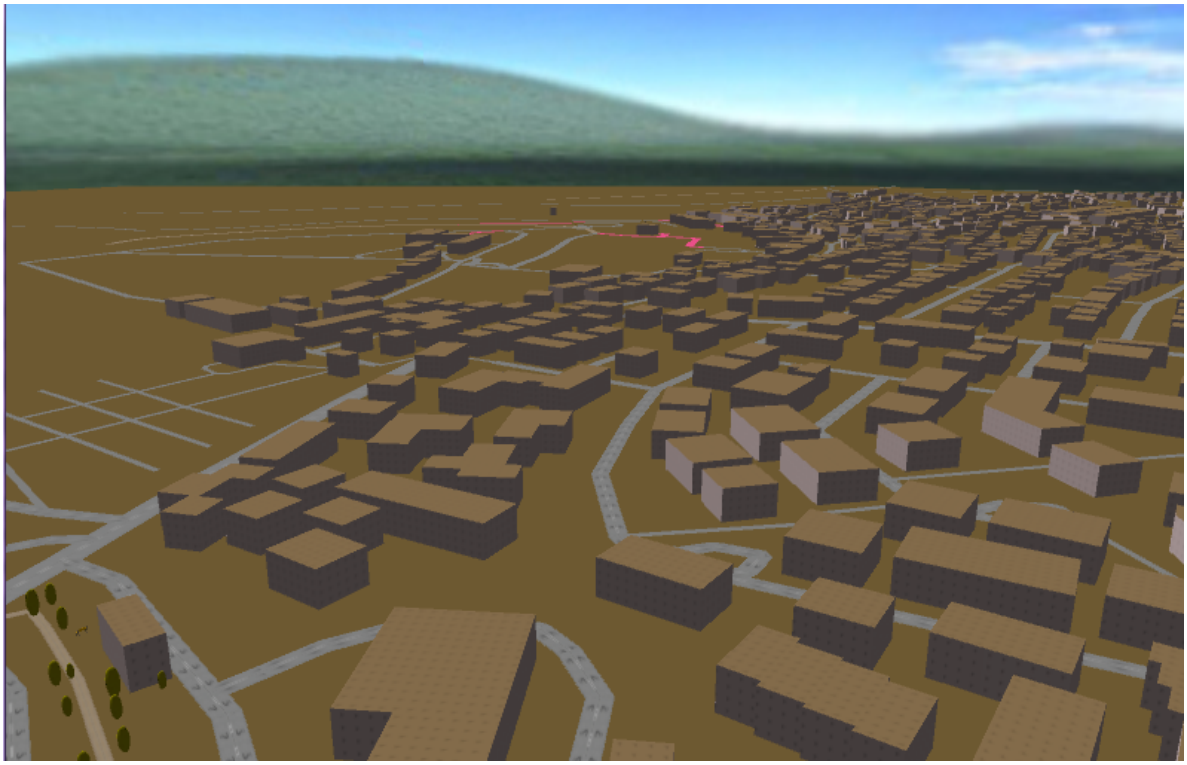

Abbildung 4.11: Musberg gerendert mit Kendzi3D

4.2.3 KothicJS

Allgemeine Übersicht

KothicJS ist eine Browser-Kartenapplikation, welche sich OSM-Kartendaten von einem Fileserver lädt, diese rendert und mit Hilfe von Leaflet darstellt. KothicJS ist eine Adaption von Kothic, welches ursprünglich lokal Kartentiles mit Hilfe von Python gerendert hat, weswegen KothicJS ebenfalls Kartentiles erzeugt [Kot13b].

Wie auch andere Applikationen, die in Echtzeit Kartendaten verarbeiten, hat KothicJS Performance Probleme. Während des Erzeugens der Kartentiles stockt die Applikation, abhängig vom Browser können Verzögerungen von bis zu 10 Sekunden auftreten. Schuld daran ist, dass JavaScript nur bedingt Multithreading-fähig ist und das Erzeugen der Kartentiles sehr rechenaufwändig ist. Außerdem wird das HTML5 Canvas Element benutzt, welches bekanntermaßen nicht sehr schnell ist und die Grafikkarte nicht mit benutzen kann. Sobald aber alle Kartentiles erstellt wurden, können diese ohne Verzögerung angezeigt werden.

Funktionsumfang und Darstellung

KothicJS ist, abgesehen von Performance-Problemen, sehr ausgereift und weist nur gelegentlich kleine Darstellungsfehler auf. Es wird die aktuelle Version der MapCSS unterstützt, mit welcher man selbst das Aussehen der Karten bestimmen kann. Straßennamen werden entlang von Straßen eingeblendet und auch generell wirkt das Labeling sehr ausgereift. Da KothicJS zwar Kartendaten erzeugt, aber diese nicht selbst anzeigen kann, wird ein zusätzliches Programm, zum Beispiel Leaflet, benötigt. Mit diesem können, je nach Bedarf, auch zusätzlich Overlays eingeblendet werden [Kot13c].

Installation und Voraussetzungen

KothicJS benötigt technisch gesehen keine Installation. Der Anwender benötigt lediglich einen modernen Browser mit JavaScript. Es ist vorgesehen, dass man eigene Kartendaten mit Hilfe eines Fileservers bereitstellt. KothicJS ist jedoch flexibel und kann Kartendaten aus beliebigen Quellen beziehen, solange sie im vorgegebenen GeoJSON-Format sind. Theoretisch kann KothicJS also auch als Offline-Anwendung betrieben werden, wenn die entsprechenden Daten lokal vorhanden sind.

Lizenzen

KothicJS steht unter der BSD-Lizenz, welche eine Nutzung und Weiterentwicklung, auch zu kommerziellen Zwecken, gestattet. Es existiert ein GitHub-Repository, in welchem der Code offen zur Verfügung steht [Kot13d]. KothicJS wurde von nur drei Personen entwickelt, welche das Projekt auch weiterhin betreuen und gelegentlich kleine Änderungen daran vornehmen.

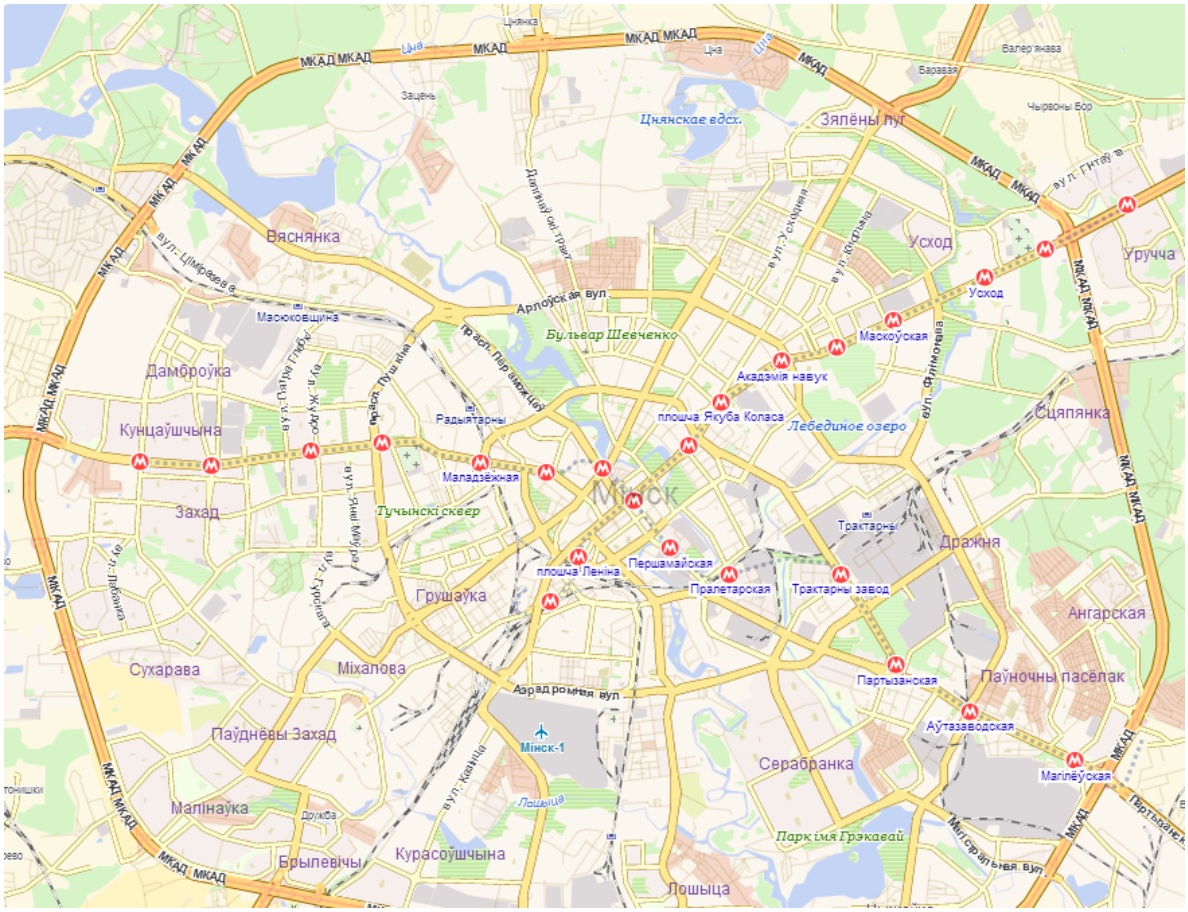

Abbildung 4.12: Moskau gerendert mit KothicJS - In Echtzeit gerendert

4.2.4 Maperative

Allgemeine Übersicht

Der Kartenrenderer Maperative ist ein Freeware-Tool mit graphischer Oberfläche und der Möglichkeit 2D, sowie 3D Karten zu erstellen.

Funktionsumfang und Darstellung

Das Kartenmaterial kann mittels Regeln angepasst werden. Automatisch bezogene Höhendaten können als Hill-Shading, Elevation Coloring oder Relief angezeigt werden. Weitere Darstellungen von Höhendaten können über Phyton-Skripte realisiert werden [Map13e].

Es ist möglich, eigene Icons, sowie Formen als Overlay einzubinden [Map13g].

Mit vorhandenen OSM-Dateien können offline Karten gerendert werden, allerdings stehen die Höhendaten nur online zum automatischen Download zur Verfügung.

Die wichtigen Straßen sind dicker dargestellt. Das Labeling orientiert sich an den Straßen und besitzt eine Kollisions-Erkennung.

Die Karten können in folgende Formate exportiert werden:

- SVG
- Bitmap
- Collada (3D Darstellung)
- Tiles

Installation und Voraussetzungen

Maperative ist für die Microsoft Betriebssysteme ab Windows XP SP3, für MacOS X, für die Linux Distributionen ab Ubuntu 10.04 und Fedora 14 verfügbar. Zur Ausführung wird Microsoft .NET 4 oder Mono benötigt. Die Installation ist mittels eines Installers realisiert. Über neue Produktversionen wird der Nutzer automatisch informiert. Diese können direkt aus der Anwendung heraus automatisch heruntergeladen und installiert werden.

Das Kartenmaterial wird online von den OSM Servern, oder von OSM-Dateien bezogen. Der Import aus Datenbanken, sowie das Hosten von Tile-Servern ist nicht möglich.

Aktuell findet eine Weiterentwicklung von Maperative statt.

Lizenzen

Maperative ist als Freeware veröffentlicht. Der Source Codes ist allerdings nicht öffentlich zugänglich. Die erzeugten Grafiken können privat, sowie kommerziell genutzt werden und unterliegen keiner Lizenz [Map13f]

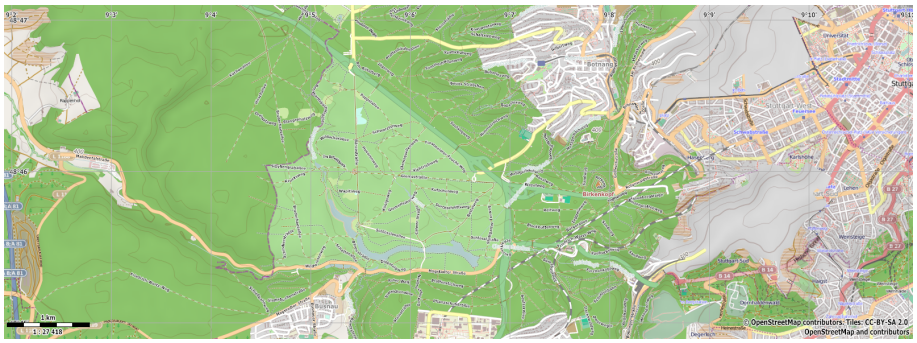

Abbildung 4.13: Stuttgart Bärensee gerendert mit Maperative. Die Höhendaten werden durch Höhenlinien und Hill-Shading dargestellt.

4.2.5 Mapnik

Allgemeine Übersicht

Mapnik ist eine API, mit der parametrierbare Karten gerendert werden können. Die API steht für verschiedene Programmiersprachen und damit auch verschiedene Plattformen zur Verfügung. Durch verschiedene Input-Plugins ist sie sehr vielfältig einsetzbar.

Funktionsumfang und Darstellung

Mapnik bietet eine API, die für C++, Python, Ruby, Java und node.js verfügbar ist. Mithilfe der API können Karten gerendert und als Datei ausgegeben werden. Dank verschiedener Plugins sind verschiedene Eingabeformate für die Kartendaten möglich. Neben Shapefiles und OSM-Dateien ist es u.a auch möglich, Daten direkt aus PostGIS Datenbanken zu beziehen [Map13j].

Zur Parametrierung der Karte ist es einerseits möglich, die Regeln direkt im Programmcode zu erstellen, andererseits kann auch ein XML-File benutzt werden. Es ist möglich, beliebig viele Layers mit jeweils beliebig vielen Regeln übereinander zu zeichnen. Dabei können Overlays in Form von geometrischen Figuren und Bildern über die Karte gelegt werden [Map13k]. Auch Labeling wird unterstützt und sieht sehr gut aus, da die Labels immer in die Straße gezeichnet werden. Eine Unterstützung für die Darstellung von Höhendaten gibt es nicht.

Die Ausgabe kann in PDF-, PS-, PNG-, JPEG-, TIFF- und SVG-Dateien erfolgen [Map13i].

Installation und Voraussetzungen

Dank der gut beschriebenen API [Map13h] ist es einfach, Mapnik in eine eigene Anwendung einzubinden. Der Sourcecode kann einfach heruntergeladen werden und steht für Windows, Linux und iOS zur Verfügung. Es gibt eine Beispielanwendung um zu sehen, was Mapnik kann.

Lizenzen

Mapnik steht unter der LGPL-Lizenz. Der Sourcecode ist öffentlich kostenlos zugänglich.

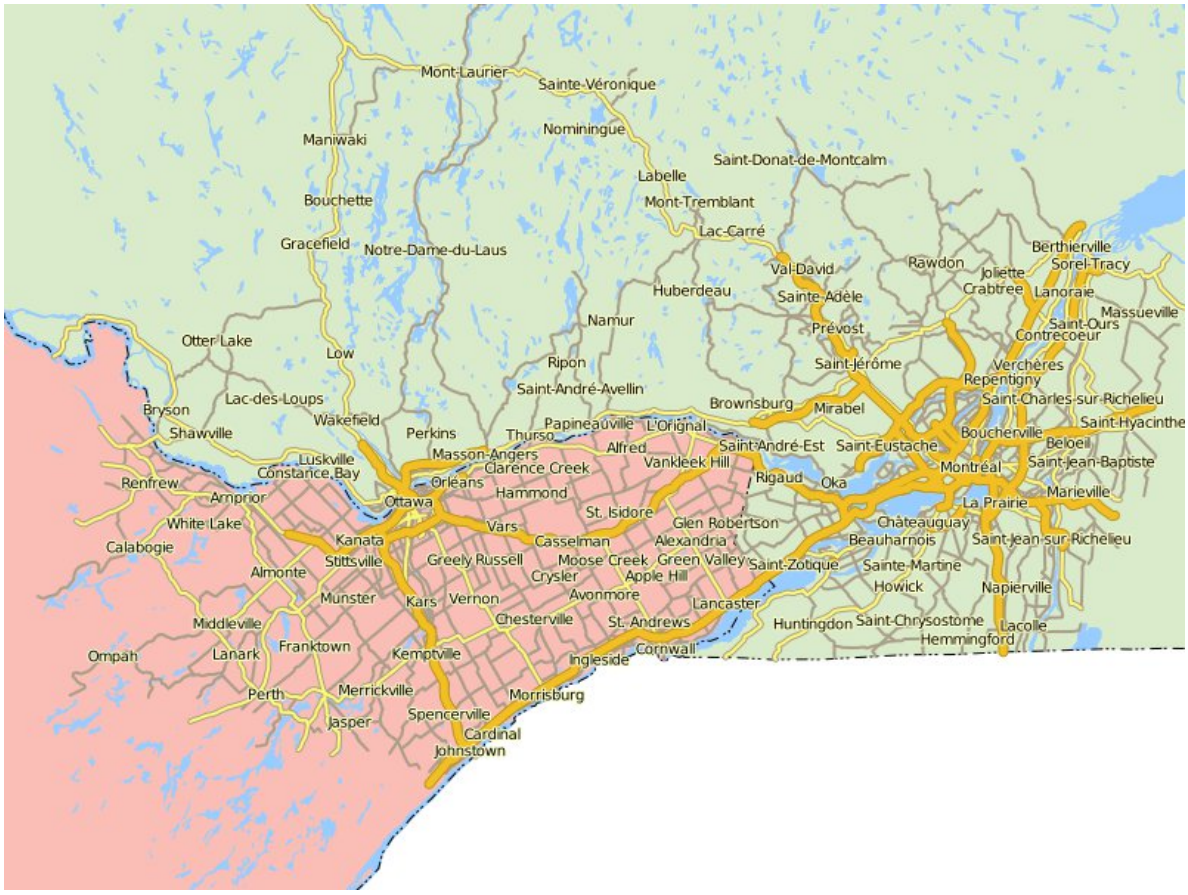

Abbildung 4.14: Kanada gerendert mit Mapnik

4.2.6 MapOSMatic

Allgemeine Übersicht

MapOSMatic ist eine Webapplikation, die dazu dient, druckbare Stadtpläne zu erstellen. Die Anwendung kann jedoch auch lokal und offline genutzt werden [Map13l].

Funktionsumfang und Darstellung

Die Karten werden um ein Raster und eine Auflistung sämtlicher Straßen im gerenderten Gebiet ergänzt, sodass sich Straßen leichter auf dem Plan finden lassen. Dabei können verschiedene Einstellungen vorgenommen werden. Zunächst kann ein Gebiet anhand des Stadtnamens herausgesucht werden. Alternativ lässt sich ein Gebiet auch durch die Eingabe von Koordinaten festlegen. Nun können die Position des Straßenverzeichnisses, sowie der Darstellungsstil der Karte aus einigen vordefinierten Möglichkeiten ausgewählt werden. Außerdem kann die Größe, die Ausrichtung der Karte, sowie die zu benutzende Sprache festgelegt werden. Wird eine Stadtgrenze erkannt, werden Elemente außerhalb dieser Stadt ausgegraut. Nach ungefähr einer halben Minute steht der Plan als PDF zum Download bereit. Bereits erzeugte Pläne werden online gespeichert und müssen bei einer erneuten Anfrage nicht erneut gerendert werden, sondern stehen direkt zum Download bereit. Da das Rendern sehr aufwendig ist, wird eine Queue fürs Rendern benutzt. Es ist also auch möglich, dass man länger auf den Plan warten muss, jedoch ist dies bei unseren Tests nie vorgekommen.
MapOSMatic nutzt OpenStreetMap-Daten zum Rendern, welche nach Aussage der Betreiberseite stets aktuell sind [Map13l]. Es ist auch möglich, die MapOSMatic-Softwarekomponenten offline zu installieren, um nicht den Webservice nutzen zu müssen. Es ist dann eine PostgreSQL Datenbank mit PostGis-Erweiterung nötig, auf die die Software zugreifen kann [Map13l]. Wie das genau funktioniert konnten wir nicht herausfinden, da das Wiki zum Zeitpunkt der Erstellung offline war.

Installation und Voraussetzungen

Benutzt man den Webservice, ist keine Installation nötig. Die einzige Voraussetzung ist JavaScript im Browser.

Lizenzen

MapOSMatic ist komplett kostenfrei und steht unter der AGPLv3-Lizenz. Der Quellcode ist öffentlich zugänglich.

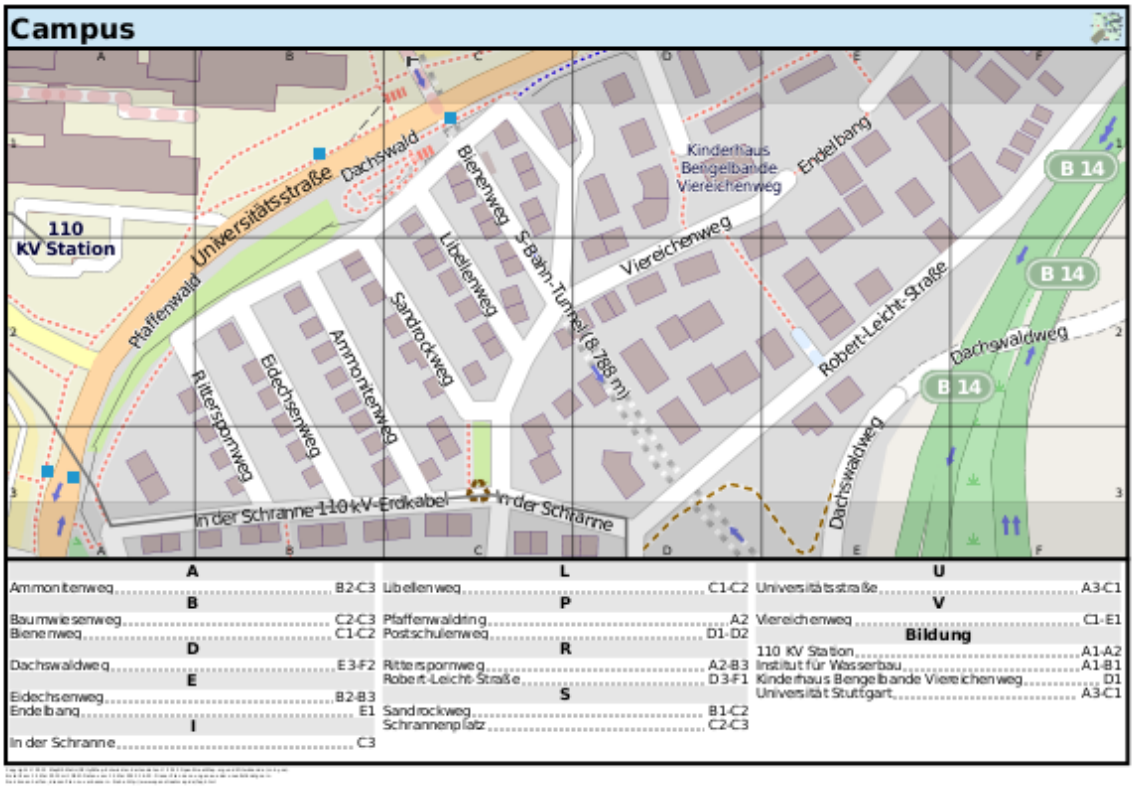

Abbildung 4.15: Campus Vaihingen gerendert mit MapOSMatic

4.2.7 Mapsforge

Allgemeine Übersicht

Mapsforge ist ein Studentenprojekt, welches an der Freien Universität Berlin um das Jahr 2008 entstanden ist. Ziel war es, einen parametrierbaren Offline-Renderer für Android-Geräte zu erstellen. Das Projekt ist mittlerweile in einem Beta-Stadium, es wurden alle Funktionen umgesetzt und sind auch weitestgehend fehlerlos. Allerdings wird das Projekt zur Zeit nur von einem wissenschaftlichen Mitarbeiter betreut, weswegen es kaum noch Veränderungen gibt [Fre13a].

Besonders hervorzuheben ist die einfache Einbindung, da sich die Mapsforge-API an die Google Maps-Api anlehnt. Da Offline-Karten genutzt werden, wird theoretisch auch kein Server für den Betrieb benötigt. Als Datenquelle werden die OSM-Daten verwendet. Diese werden mit Hilfe eines bereitgestellten Programmes so vor verarbeitet, dass sie weniger Platz benötigen und auf schnelle Zugriffszeiten optimiert sind. Dennoch kann Mapsforge bei der Darstellungsgeschwindigkeit nicht mit statischen Kartenprogrammen mithalten.

Funktionsumfang und Darstellung

Mapsforge ist an und für sich nur eine API, es existiert jedoch auch eine Beispielanwendung, welche als Anleitung zum Einbinden dient. Mapsforge kümmert sich um das Erstellen und Darstellen der Karten, zudem können Overlays eingeblendet werden. Besonders hervorzuheben ist die Möglichkeit, Karten mit Hilfe einer XML nach eigenen Wünschen zu gestalten. Die Darstellung funktioniert relativ gut, gelegentlich gibt es kleine Darstellungsfehler.

Installation und Voraussetzungen

Um Mapsforge nutzen zu können, ist die Einbindung in eine App bzw. deren Installation notwendig. Zusätzlich müssen Kartendaten vorverarbeitet und bereit gestellt werden. Abhängig davon, welcher Kartenausschnitt verfügbar sein soll, ändert sich die Größe der Anwendung. Für Berlin in hoher Auflösung werden ca. 15 MB benötigt, was verglichen mit anderen Anwendungen relativ wenig ist.

Lizenzen

Die API steht unter der LGPL Lizenz und darf demnach kostenlos genutzt werden. Der Quellcode steht offen zur Verfügung und darf auch bearbeitet werden. Alle Änderungen werden von einem Mitarbeiter der Freien Universität Berlin geprüft, welcher zur Zeit auch für die Weiterentwicklung des Projekts zuständig ist [Fre13b].

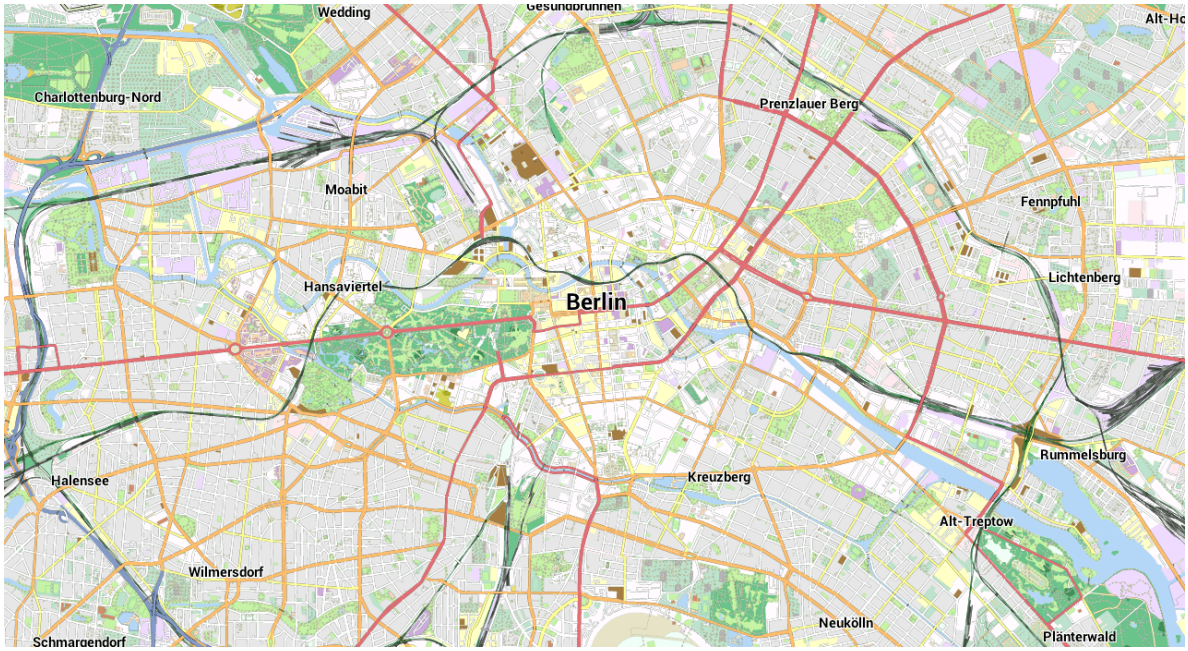

Abbildung 4.16: Berlin gerendert mit Mapsforge

4.2.8 Mapweaver

Allgemeine Übersicht

Mapweaver ist ein Programm zum Rendern von OSM-Daten. Die Ausgabe erfolgt als SVG- oder PDF-Datei. Die Darstellung kann mithilfe einer Regel-Datei parametriert werden. Das Programm ist in Perl geschrieben und ist für die Ausführung unter Linux gedacht.

Funktionsumfang und Darstellung

Eine interessante Funktion, die Mapweaver von anderen Renderern unterscheidet, ist, dass sich Mapweaver die erforderlichen Daten selbst herunterladen kann [Map13p]. Man kann Stadtnamen angeben, die dann auf OpenStreetMaps gesucht und heruntergeladen werden. Alternativ ist es auch möglich, die Ausgabe aus einer OSM-Datei offline zu erzeugen. Da keine extra Datenstruktur aufgebaut werden muss erfolgt das Rendern sehr schnell. Ein Kartenausschnitt von ca. 3x3km ist mit einem durchschnittlichen PC in weniger als 10 Sekunden gerendert und in einer Ausgabedatei gespeichert. Beim Aufruf des Renderprogramms können diverse Parameter angegeben werden, mit denen die Ausgabe beeinflusst werden kann. So kann eine Legende eingezeichnet werden, oder die Größe der Labels eingestellt werden. Regeln fürs Rendern werden in einer einfach zu editierenden CSV-Datei gespeichert. Zu jeden Objekt, das einen bestimmten Tag enthält, können Darstellungsparameter definiert werden. Es können diverse Farbwerte gesetzt werden und es kann angegeben werden, welcher Wert fürs Labeling verwendet werden soll. All diese Werte können abhängig von der Zoomstufe definiert werden. Mapweaver beherrscht das Labeling von Objekten sowie das Anzeigen von Icons auf der Karte. Es können diverse Höheninformations-Dateien verwendet werden, um entsprechende Daten in die Karte einzuzeichnen [Map13p]. Der subjektive Eindruck der Kartendarstellung ist sehr gut

Installation und Voraussetzungen

Es gibt bisher keinen Installer für Mapweaver. Für die Einrichtung muss deshalb zunächst ein SVN-Verzeichnis geklont werden [Map13q]. Nach dem manuellen Installieren einiger benötigter Dependencies kann das Programm direkt genutzt werden.

Lizenzen

Für Mapweaver wurde bisher nicht angegeben, unter welcher Lizenz es genutzt werden kann. Der Quellcode ist jedoch öffentlich kostenlos zugänglich [Map13r].

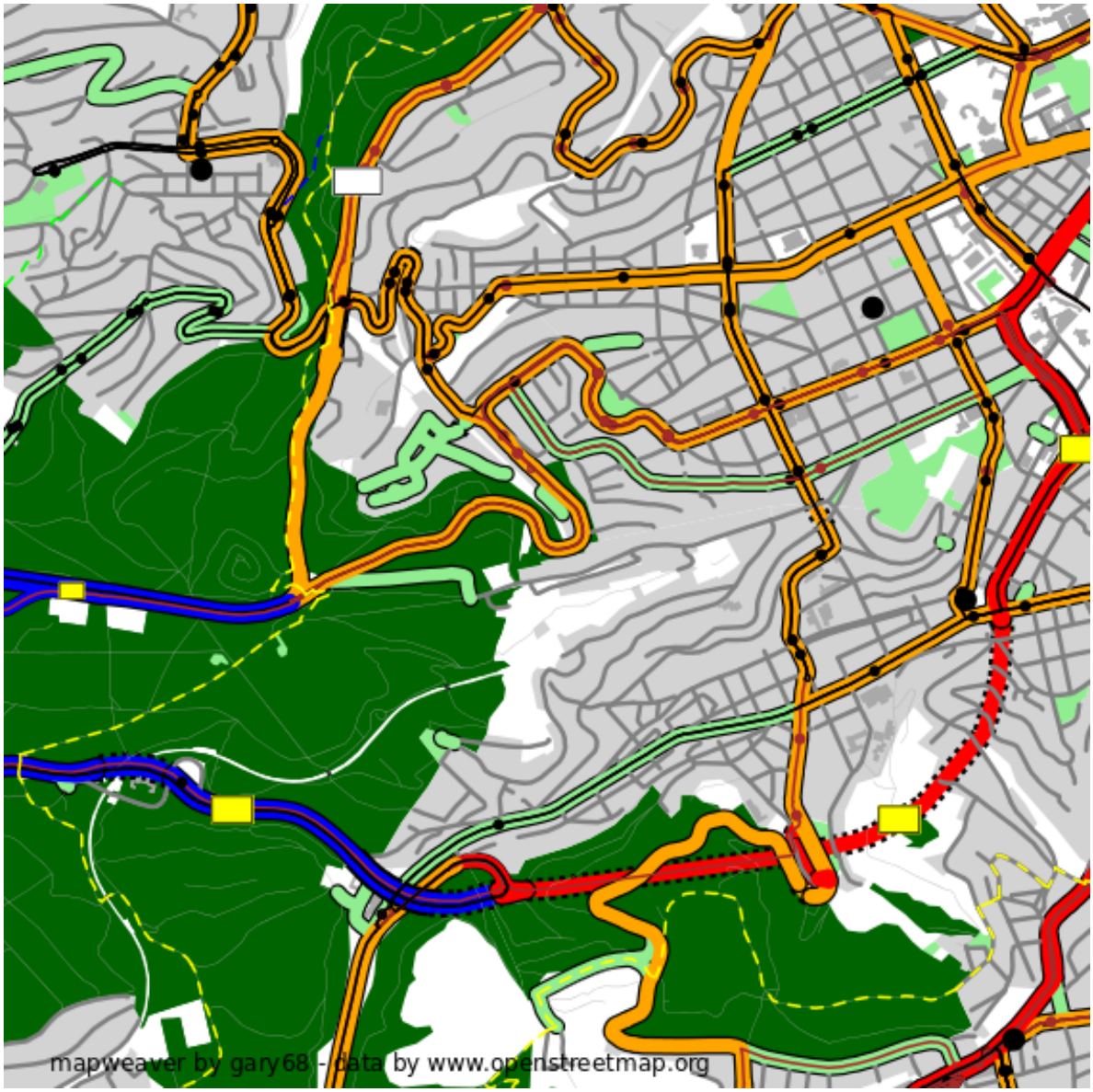

Abbildung 4.17: Stuttgart gerendert mit Mapweaver - In der Standardkonfiguration

4.2.9 Memphis

Allgemeine Übersicht

Memphis ist eine kostenlose Konsolenanwendung zum Erstellen von Karten im PNG Format und unter Linux nutzbar. Aktuell findet keine Weiterentwicklung von Memphis statt.

Funktionsumfang und Darstellung

Memphis erstellt sehr einfach gehaltene Karten die mit Hilfe von Regeln parametriert werden können. Die Karten enthalten allerdings einige Fehler bei Gegenden die Sonderbauten enthalten, wie z.B. dem Flughafen oder dem Hauptbahnhof in Stuttgart. Ein Labeling ist nicht implementiert. Die Kartenansicht ist nicht orthogonal, sondern leicht schräg dargestellt. Des weiteren bietet der Hersteller eine GObject API an [Ohl13]. Für eine Tile-Server Anwendung ist beim Hersteller eine Demo verfügbar.

Installation und Voraussetzungen

Memphis wurde im Rahmen dieser Fachstudie für Linux kompiliert. Eine Ausführung für Windows oder MacOS wird vom Hersteller nicht garantiert.

Lizenzen

Die Nutzung von Memphis ist kostenlos. Der Source Code ist unter LGPL-2.1+ Lizenz veröffentlicht.

Der Kartenrenderer benötigt eXpat, Cairo und Glib [Ohl13].

Abbildung 4.18: Stuttgart Bärenseen gerendert mit Memphis. Es ist kein Labeling implementiert.

4.2.10 OpenCycleMap

Allgemeine Übersicht

OpenCycleMap ist ein Webservice, der vorgerenderte Tiles für unterschiedliche Anwendungszwecke als PNGs zur Verfügung stellt. Die dafür benötigten Daten stammen von OpenStreetMap. Die Tiles können in Anwendungen eingebunden werden [Ope13i], die Leaflet, OpenLayers oder CloudMade Web Maps zur Darstellung von Karten verwenden.

Funktionsumfang und Darstellung

Aktuell gibt es drei verschiedene Kartendarstellungen:

- OpenCycleMap
- Transport
- Landscape

Die OpenCycleMap ist – wie der Name bereits vermuten lässt - an die Bedürfnisse von Fahrradfahrern zugeschnitten. Auf hoher Zoomstufe sollen Radwege übersichtlich dargestellt werden, sodass man erkennen kann, welche Wege sich zum Fahrradfahren eignen. Auf den niedrigeren Zoomstufen will OpenCycleMap eine gute Übersicht schaffen, um Radtouren planen zu können [Ope13j]. Außerdem sind POIs, die für Fahrradfahrer interessant sind, also bspw. Fahrradshops, eingeblendet. Höheninformationen werden als Höhenlinien dargestellt und stammen von der NASA.

Die Transport-Kartenansicht ist dazu gedacht, öffentliche Verkehrsmittel auf der ganzen Welt übersichtlich darzustellen. Es werden Haltestellen, sowie Fahrrouten von öffentlichen Verkehrsmittel angezeigt. Straßen und Gebäude rücken deutlich in den Hintergrund.

Die Ladscape-Kartenansicht soll Landschaften in den Vordergrund rücken und Städte oder Straßen in den Hintergrund wandern lassen. So werden Höhenlinien sehr deutlich angezeigt und Berggipfel eingezeichnet.

Interessant ist, dass man sich durch den Betreiber der Seite eigene Kartenansichten erstellen lassen kann, die dann auch auf dessen Seite zur Verfügung gestellt werden [Ope13k].

Installation und Voraussetzungen

Die Einbindung in eigene Anwendungen ist dank der Leaflet-Konformität sehr einfach. Der Server stellt Tiles als PNG-Dateien zu Verfügung. Die Dateien sind nach Koordinaten banannt. So muss bei eigenen Anwendungen mit Leafletintegration nur der Servername von OpenCycleMaps angegeben werden.

Lizenzen

OpenCycleMap benutzt eine eigene Lizenz [Ope13l]. Die Nutzung ist bis zu einem nicht näher definierten Volumen sowohl für den privaten, als auch den kommerziellen Einsatz kostenfrei. Bei Überschreitung des Volumens müssen monatliche Zahlungen in Höhe von mindestens £300 getätigt werden [Ope13l].

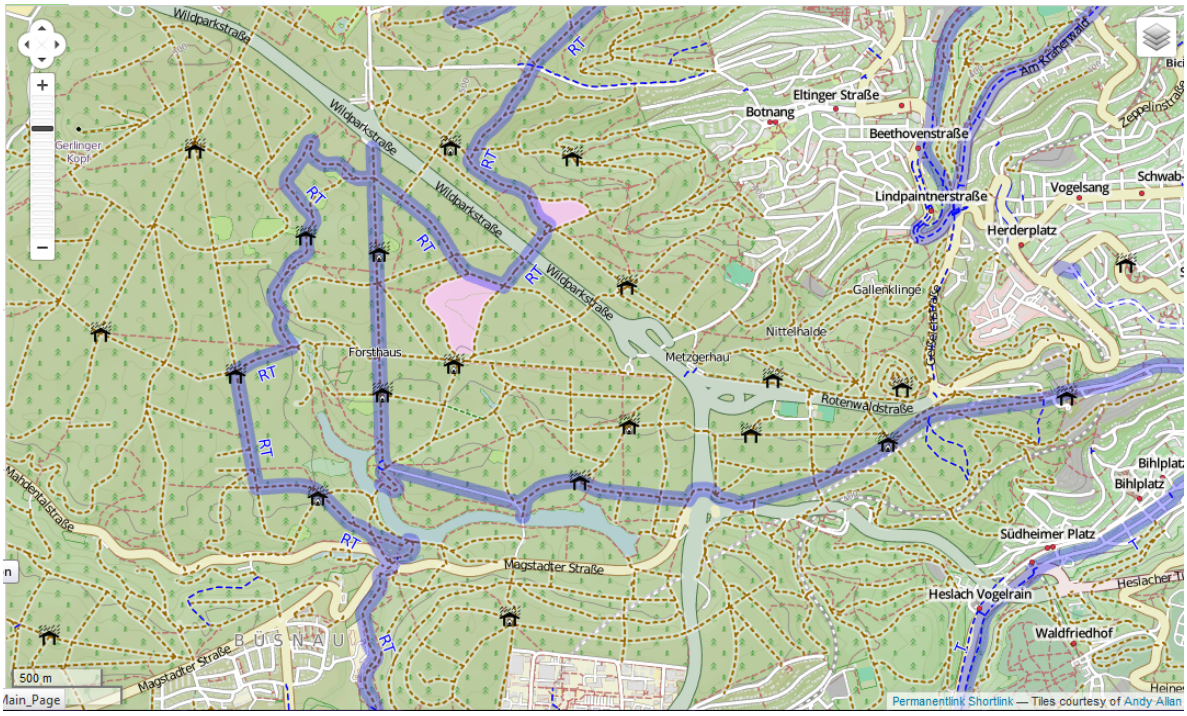

Abbildung 4.19: Stuttgart Bärensee gerendert mit OpenCycleMap

4.2.11 OpenStreetBrowser

Allgemeine Übersicht

OpenStreetBrowser ist eine Applikation, die im Browser läuft und OpenStreetMap-Daten darstellt, sowie eine Punkt-zu-Punkt-Navigation anbietet. Zusätzlich ist es möglich, verschiedene Overlays auf der Karte einzublenden, wodurch sich OpenStreetBrowser von der Kartenanzeige auf OpenStreetMap.org unterscheidet.

Funktionsumfang und Darstellung

Es gibt verschiedene Arten von Overlays, die OpenStreetBrowser anzeigen kann:

- eigene Position auf der Karte
- die Route der Routenberechnung
- diverse Informationen zu öffentlichen Verkehrsmitteln (Fahrstrecken, Haltestelle,...)
- POIs

Zur Positionsbestimmung benutzt OpenStreetBrowser die entsprechende Funktion des Browsers. Zur Anzeige von POIs muss zuvor die entsprechende Kategorie und ggf. Unterkategorie ausgewählt werden (Bsp.: Transport -> Öffentlicher Verkehr -> Haltestellen).

Eine weitere Funktion von OpenStreetMap ist "Was ist hier?". Man wählt dazu einen Punkt auf der Karte aus und wählt die entsprechende Option. Nun zeigt OpenStreetBrowser alle Objekte aus den OpenStreetMap-Daten an, die diese Position beinhalten. Wählt man eines dieser Objekte aus, so färbt OpenStreetBrowser das Objekt ein, um dessen Ausmaße zu erkennen. Dies funktioniert auch für größere Gebiete wie beispielsweise „Stuttgart-West“. Wählt man ein Gebäude aus, kann OpenStreetBrowser auch dort ansässige Unternehmen anzeigen. Diese Funktionen sind hauptsächlich für Anwender gedacht, die die Karte untersuchen und bearbeiten wollen [Ope13d].

Um die Punkt-zu-Punkt Navigation zu nutzen, müssen ein Start- und ein Endpunkt und ggf. Zwischenstopps auf der Karte gewählt werden. Nun wird die Route auf der Karte dargestellt und in der Seitenleiste sind Navigationsanweisungen zu finden. Die Routenberechnung übernimmt die externe Seite Cloudmade [Ope13d].

Die Darstellung entspricht weitgehend der Darstellung auf OpenStreetMap.org. Es ist nur eine 2D Darstellung möglich. Die Karte kann mit vier verschiedenen vorgefertigten Farbschemata angezeigt werden. Zur Steuerung der Karte können entweder die auf der Karte oben links gezeigten Navigationselemente oder die üblichen Mausgesten verwendet werden.

Da die Tiles erst bei der jeweils ersten Anfrage vom Server gerendert werden, kann es in entlegenen Gebieten lange dauern, bis die Karte angezeigt wird. Normalerweise ist die Anzeige jedoch sehr schnell und flüssig. Laut OpenStreetBrowser sind die Kartendaten stets aktuell und decken die gesamte Welt ab [Ope13d].

OpenStreetBrowser bietet ein API an, um einen Kartenausschnitt mit definierten Overlays kostenlos auf einer Webseite anzeigen zu können.

Installation und Voraussetzungen

Da es eine Webanwendung ist, gibt es außer einem Browser mit JavaScript-Unterstützung keine weiteren Voraussetzungen.

Lizenzen

Leider gibt es keine Angabe zur verwendeten Lizenz. Der Quellcode ist jedoch öffentlich zugänglich. Die Nutzung ist kostenlos.

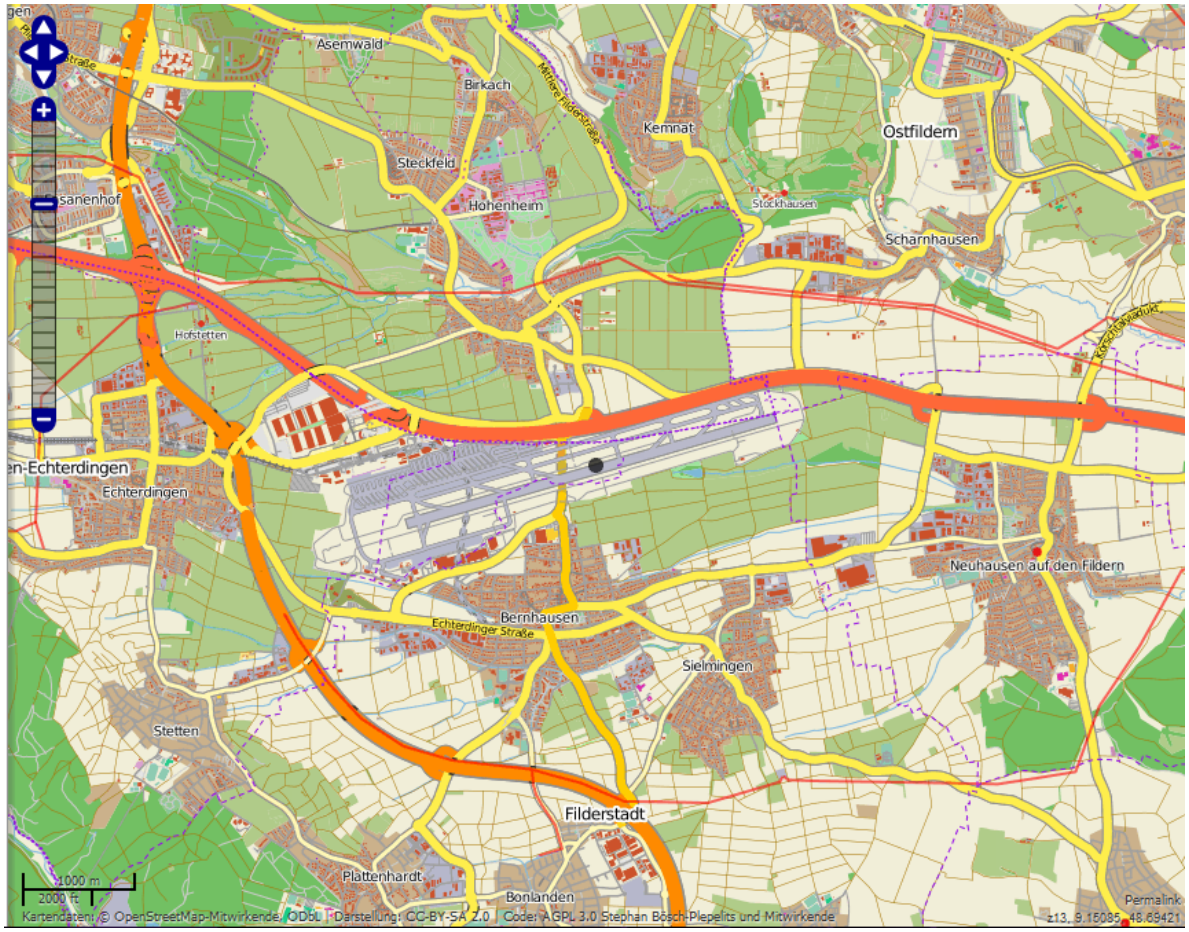

Abbildung 4.20: Flughafen Stuttgart gerendert mit OpenStreetBrowser

4.2.12 Osmand

Allgemeine Übersicht

Osmand ist eine kostenlose Android-App, die Karten anzeigen und Routen berechnen kann. Die Kartendaten stammen von OpenStreetMap und werden auf dem Gerät gespeichert, wodurch Osmand offline verwendet werden kann.

Funktionsumfang und Darstellung

Um die Kartendaten anzuzeigen, müssen diese zuvor heruntergeladen werden. Dies geschieht direkt aus der App heraus. Es kann ausgewählt werden, welche Region, oder welches Land heruntergeladen werden soll. Weiterhin gibt es eine Weltübersicht, die die größten Städte aller Länder enthält, um einen Überblick auf einer hohen Zoomstufe zu erhalten. Die Weltübersicht zusammen mit den Kartendaten für Baden-Württemberg benötigen ca. 250 MB auf dem Gerät. Um eine Sprachausgabe bei der Navigation verwenden zu können, muss zusätzlich noch das Sprachpaket heruntergeladen werden, was jedoch nur ca. 1MB groß ist. Alle unterstützten Länder außer Deutschland können in einem kompletten Paket heruntergeladen werden - Für Deutschland sind die Pakete in Bundesländer unterteilt. Dies liegt vermutlich daran, dass in der kostenlosen Version von Osmand insgesamt nur 10 Pakete auf dem Gerät gespeichert werden können. Es ist also nicht möglich, mit der kostenlosen Version die Kartendaten von ganz Deutschland zu nutzen, da es für alle Bundesländer 16 Pakete wären. Außerdem werden die Sprachpakete und die Weltübersicht ebenfalls mitgezählt. In der kostenpflichtigen Version, bestehen diese Beschränkungen nicht. Außerdem dieser Beschränkung ist der Funktionsumfang der kostenlosen und der kostenpflichtigen Version identisch. Die kostenpflichtige Version kann für 5,99 in Google Play bezogen werden.

Die Navigationsfunktionen von Osmand sind sehr vielfältig. Es lässt sich auswählen, ob man die Route für Fußgänger, Autofahrer oder Fahrradfahrer berechnen möchte. Außerdem lassen sich z.B. Mautstraßen oder Autobahnen meiden. Durch ein Plugin-System ist es möglich, Osmand um weitere Funktionen zu erweitern. So können auch Önline-Karten"verwendet werden. Dann ist es nicht mehr nötig, alle Karten auf dem Gerät vorzuhalten. Außerdem ist es z.B. auch möglich OSM-Daten direkt aus Osmand heraus zu bearbeiten. Osmand stellt eine umfangreiche Suche zur Verfügung. Neben der Adresssuche und der Suche nach Koordinaten, kann auch nach POIs einer bestimmten Kategorie gesucht werden. Außerdem gibt eine Suche für Öffentliche Verkehrsmittel

Osmand kann Overlays anzeigen. Dazu gehören POIs, aber z.B. auch das aktuelle Verkehrsaufkommen. Labeling wird ebenfalls unterstützt, das Anzeigen von Höhendaten jedoch nicht. Osmand beinhaltet mehrere Kartenansichten, die bestimmte Details der Karte hervorheben sollen. Eine freie Anpassung der Karte beispielsweise mittels Regeln gibt es nicht. Obwohl Osmand noch im Betastadium ist, macht es einen sehr guten Eindruck. Es hat sehr mächtige Funktionen und ist trotzdem einfach zu bedienen. Leider bietet Osmand keine API an.

Installation und Voraussetzungen

Die Installation ist sehr simpel, da Osmand in Google Play gefunden werden kann. Außerdem findet sich Osmand auch als kompilierte Datei auf der Webseite von Osmand [Osm13c]. Es wird mindestens Android 1.6 vorausgesetzt [Osm13d].

Lizenzen

Osmand ist Opensource und steht unter einer eigenen Lizenz.

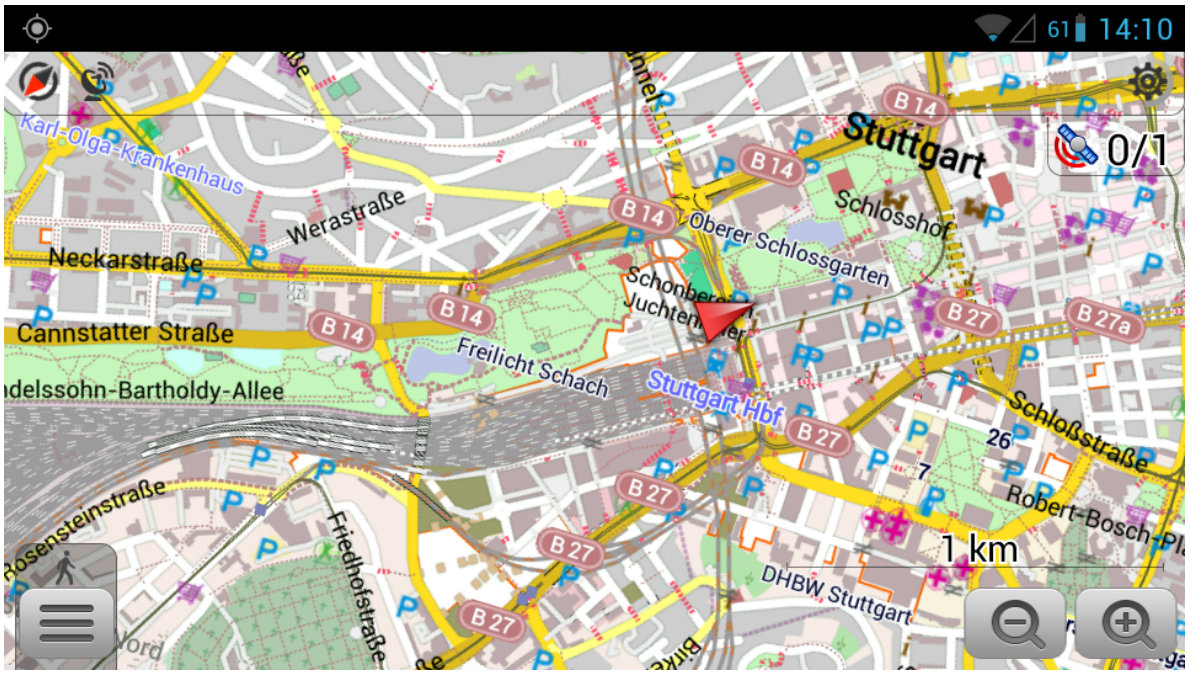

Abbildung 4.21: Stuttgart gerendert mit Osmand

4.2.13 Osmarender

Allgemeine Übersicht

Osmarender ist ein XSLT Tool, dass mittels eines XSLT Prozessors ausgeführt wird. Der XSLT Prozessor wandelt die OSM Kartendaten zusammen mit dem XML Transformationsbeschreibung in eine Grafik um. Diese sind in den gängigen Browsern, sowie als einzelne Anwendung verfügbar und somit auf allen gängigen Systemen und auch in einer Server-Anwendung einsetzbar. Das Tool nutzt OSM-Kartenmaterial und wird als Freeware veröffentlicht. Seit März 2012 wird Osmarender nicht mehr gewartet [Ope13p].

Funktionsumfang und Darstellung

Osmarender erstellt mit Hilfe von Regeln SVGs. Weiterhin ist es möglich auch eine interaktive verschiebbare Karte mit Zoomstufen zu erstellen, die in ihrer Grundfunktion der von openstreetmap.org ähnelt. Osmarender eignet sich zum Rendern von Karten lokal auf einem Rechner, sowie als Teil einer Webanwendung. Die Karten können durch Overlays mit eigenen Symbolen und Layern ergänzt werden [Ope130]. Eine Verarbeitung von Höhendaten ist nicht möglich. Das Labeling der Straßen folgt dem tatsächlichen Verlauf der Straße.

Installation und Voraussetzungen

Osmarender liegt in verschiedenen Ausführungsvarianten, Weiterentwicklungen und Implementierungsformen vor.

Inital liegt Osmarender als XSL-Anwendung vor, die mit einem XSLT-Prozessor oder Browser ausgeführt wird. Die einzelnen benötigten Dateien müssen manuell heruntergeladen werden und mit einer geeigneten Plattform ausgeführt werden.

Als eine Weiterentwicklung wurde auch ein Frontend veröffentlicht, mit dem Osmarender über eine GUI ausgeführt werden kann [Ope13m].

Eine äquivalente Perl-Implementierung von Osmarender ist ebenfalls verfügbar.

Lizenzen

Osmarender wurde unter der GPL Lizenz veröffentlicht. Der Source-Code ist frei zugänglich [Ope13n].

Abbildung 4.22: London gerendert mit Osmarender

4.2.14 SMRender

Allgemeine Übersicht

SMRender ist ein Kartenrenderer, der offline rendert. Gedacht ist SMRender dafür, Seekarten zu rendern und zu drucken. Dank seines regelbasierten Renderns ist er jedoch vielseitig einsetzbar. SMRender wurde für Linux entwickelt, läuft aber mit Cygwin auch auf Windows [SMR13a].

Funktionsumfang und Darstellung

SMRender benutzt OSM-Dateien als Input und erzeugt daraus entweder PNG- oder PDF-Dateien. Die Benutzung erfolgt durch ein Konsolenprogramm, das mit Parametern ausgeführt werden muss. Neben den Ein- und Ausgabedateien muss zusätzlich noch eine Regel-Datei angegeben werden. Diese ist eine XML-Datei, die Tags verschiedene Renderregeln zuordnet. Dabei können z.B. Farbe oder Strichart festgelegt werden. Es ist auch möglich geometrische Figuren und Bilder einzuzeichnen. Auch ein Labeling wird unterstützt.

Angenehm ist, dass man angeben kann, welcher Kartenausschnitt gerendert wird. Es ist also möglich, eine OSM-Datei zu verwenden, die ganz Deutschland umfasst und durch die Eingabe von Koordinaten nur beispielsweise den Raum Stuttgart zu rendern. Dabei wird neben der normalen Koordinatendarstellung auch das nautische Format als Eingabe akzeptiert.

Eine weitere Funktion, die jedoch hauptsächlich für das Rendering von Seekarten gedacht sein dürfte, ist die Möglichkeit,auf Wunsch Gitterlinien mit Koordinaten einzublenden. Auch diese Darstellung kann durch die Regeldatei angepasst werden [SMR13c]. Da SMRender dafür gedacht ist, die gerenderten Karten auszudrucken erfolgt die Ausgabe in einem Din-A Seitenverhältnis und mit einer für den Druck geeigneten Auflösung von 300dpi. Diese Einstellungen lassen sich jedoch durch Parameter ändern. Zusätzlich ist es möglich, die Hintergrundfarbe der Karten festzulegen - also für den Bereich, in dem nichts gezeichnet wird.

Die Karten werden schnell gerendert und sehen gut aus. Jedoch ist das setzen der ganzen Parameter etwas mühsam. Es wäre angenehmer, wenn es dafür eine Konfigurationsdatei geben würde.

Installation und Voraussetzungen

Die Einrichtung ist recht einfach. Da es jedoch keinen Installer gibt, muss vieles von Hand gemacht werden. Zunächst muss ein Archiv heruntergeladen und entpackt werden. Dies

beinhaltet den Quellcode, sowie einige Regeldateien. Um diese benutzen zu können muss ein zusätzliches Archiv mit den benötigten Icons heruntergeladen werden. Nun müssen noch das Paket libcairo heruntergeladen und installiert werden. Letztendlich kann SMRender kompiliert und ausgeführt werden. Da die mitgelieferten Regeldateien dafür gedacht sind, Seekarten darzustellen, ist zusätzlicher Aufwand nötig, diese dahingehend anzupassen, dass sie für die Darstellung von Städten geeignet sind.

Lizenzen

Die Nutzung ist kostenlos und der Quellcode ist öffentlich zugänglich [SMR13b]. Er steht unter der GPLv3-Lizenz.

Abbildung 4.23: Flughafen Stuttgart gerendert mit SMRender - Mit Standardregeln

4.3 Nicht lauffähig

4.3.1 Cartagen

Allgemeine Übersicht

Cartagen ist eine Echtzeit-Kartenrender-Applikation, die kostenlos im Browser zur Verfügung steht.

Cartagen wurde ursprünglich als Ein-Mann Projekt von Jeffrey Warren entwickelt, welcher Mitarbeiter am MIT Media Lab war [MIT13b]. Seit etwa 2011 gibt es kaum noch Updates, die Internetseite steht jedoch weiterhin zur Verfügung. Cartagen versucht, anders als viele aktuelle Kartenrenderer, Karten clientseitig zu rendern. Dabei werden lediglich die Karteninhalte vom Server übertragen und dann erst auf dem PC in Bilder umgewandelt. Das hat den Vorteil, dass die Kartendaten individuell angepasst dargestellt werden können. Der Nachteil ist aber, dass oft viel mehr Daten übertragen werden müssen und das Errechnen der Kartendaten unter Umständen lange dauert. So dauert das Anzeigen eines Kartenausschnitts zwischen 1 - 3 Sekunden, teilweise auch länger.

Funktionsumfang und Darstellung

Insgesamt ist das Projekt nicht sehr ausgereift. Oftmals werden keine neuen Kartenausschnitte nachgeladen, weil die Applikation abstürzt, ohne eine Meldung auszugeben. Zoomen funktioniert nur bedingt, auch hierbei stürzt die Applikation öfters ab. Insgesamt läuft Cartagen nicht flüssig, sondern eher stockend. Laut eigenen Angaben ist die Darstellung von Echtzeitdaten (zum Beispiel Menschenmengen während einer Demonstration) möglich, allerdings gibt es keine Schnittstelle dafür.

Bemerkenswert ist die sehr reiche Darstellungsvielfalt. Es gibt mehrere Beispiele für Darstellungsvarianten, die sich von traditionellen Kartendarstellungen stark unterscheiden. Ein Beispiel ersetzt Gebäude aus der realen Welt mit Darstellungen aus einem Computerspiel, wodurch das hier dargestellt London auf einmal mittelalterlich anmutet [MIT13a].

Installation und Voraussetzungen

Benutzer der Applikation benötigen lediglich einen modernen Browser, welcher JavaScript unterstützt. Wenn man selbst Kartendaten bereitstellen möchte, muss man diese auf einem Fileserver bereitstellen, auf den Cartagen direkt zugreift. Dadurch entfällt die Aufgabe, andauernd einen Server im Hintergrund am Laufen zu haben. Dafür entsteht aber unter Umständen ein größerer Platzverbrauch als in einer Datenbank.

Lizenzen

Cartagen steht unter der MIT License und darf ohne Einschränkungen genutzt und weiterentwickelt werden, auch zu kommerziellen Zwecken. Der Code steht in einem Git-Repository zur Verfügung und darf bearbeitet werden, allerdings hat es seit einiger Zeit keine aktive Wartung mehr gegeben [MIT13c].

Abbildung 4.24: London gerendert mit Cartagen mit World of Warcraft-Gebäuden

4.3.2 Kosmos

Allgemeine Übersicht

Der Kartenrenderer Kosmos ist die Vorgängerversion von Maperative und wird nicht mehr weiterentwickelt [Ope13h]. Seid der Umstellung der OSM Karten auf 64-Bit-Integer können mit Kosmos keine aktuellen OSM-Dateien mehr geöffnet werden.

Funktionsumfang und Darstellung

Der Karten können mittels Regeln parametrisiert werden. Höhendaten werden durch Relifen, Slope-Shading, oder Elevation Coloring visualisiert.

Komsmos ist in der Lage, einen Tile-Server zu hosten, um die gerenderten Karten Endnutzern zur Verfügung zu stellen.

Die Karten können in folgende Formate exportiert werden:

- Tiles für den Tile-Server (PNG)
- Bitmap

Installation und Voraussetzungen

Kosmos wird als zip Archiv angeboten und ist nach dem Entpacken sofort lauffähig. Komos benötigt Microsoft .NET 3.5 SP1, das ab Microsoft Windows XP verfügbar ist. Eine Portierung auf Linux mittels Mono ist nicht abgeschlossen worden.

Die Karten-Daten müssen im OSM-Format vorliegen. Über die Kosmos-GUI kann ein Bereich ausgewählt werden. Dieser Bereich wird anschließend heruntergeladen. Diese Einschränkung beschränkt zwangsläufig auch die Bereiche der renderbaren Karten. Die Karten-Daten können auch nicht aus einer Datenbank geladen werden. Des weiteren können GPS-Informationen von einem mobilen Gerät eingelesen und verarbeitet werden.

Lizenzen

Der Source-Code von Kosmos ist offen zugänglich und wurde unter BSD-Lizenz veröffentlicht [Ope13g].

4.3.3 Kothic

Allgemeine Übersicht

Kothic ist ein Programm, das Karten offline rendern und darstellen kann. Die Darstellung ist dabei konfigurierbar. Kothic ist in Python geschrieben und läuft auf Linux-Systemen, befindet sich jedoch noch in der Entwicklung

Funktionsumfang und Darstellung

Bevor Kothic rendern kann, muss es eine eigene Datenstruktur der Kartendaten aufbauen. Dazu gibt es ein Kommandozeilenprogramm, das diese Datenstruktur aus OSM-Dateien aufbaut. Dies dauert beispielsweise für Stuttgart auf einem durchschnittlichen Rechner ca. zwei Stunden. Alternativ ist es auch möglich, eine PostgreSQL-Datenbank zu verwenden. Durch eine Styles-Datei kann die Anzeige angepasst werden. Es ist möglich, Objekte, die einen angegebenen Tag haben, anzupassen. So können dort z.B. die Dicke, Farbe und Durchsichtigkeit von Straßen angegeben werden. Auch Texturen können eingebunden werden. Es ist möglich, On-The-Fly eine andere Style-Datei zu benutzen. Kothic unterstützt das Labeling von Straßen und die Anzeige von Icons (z.B. Parkplatzsymbolen) in der Karte. Das Labeling ist jedoch noch nicht besonders gut umgesetzt, da sich Straßennamen oft überschneiden. Einmal gerenderte Kartenausschnitte werden gecacht, wodurch das Zoomen und Verschieben der Karte flüssiger wird. Das Rendern eines Tiles, der bisher noch nicht angezeigt wurde, dauert jedoch mit z.T. über 15 Sekunden deutlich zu lang. Leider merkt man Kothic an, dass die Entwicklung noch nicht abgeschlossen ist. Es stürzt sehr oft ab, ist verhältnismäßig langsam und mehr als die Hälfte der mitgelieferten Styles funktionieren nicht bzw. bringen das Programm zum Abstürzen. Wir raten deshalb zu diesem Zeitpunkt von der Benutzung ab. Auch auf der Homepage von Kothic ist angemerkt, dass es noch nicht für die tägliche Benutzung geeignet ist.

Installation und Voraussetzungen

Es gibt keinen Installer für Kothic. Man muss sich den Quellcode aus dem öffentlich zugänglichen Git herunterladen. Danach müssen einige Abhängigkeiten manuell installiert werden, die es z.T. nur für Linux gibt. Letztendlich muss der Quellcode kompiliert und ausgeführt werden. Vor der eigentlichen Benutzung muss noch ein Import der Kartendaten erfolgen.

Lizenzen

Wir haben keine Angaben zur verwendeten Lizenz finden können. Der Quellcode ist jedoch öffentlich und kostenlos zugänglich [Kot13a].

Abbildung 4.25: Stuttgart gerendert mit Kothic

4.3.4 OpenStreetMap-3D

Allgemeine Übersicht

OpenStreetMap-3D versucht, die OSM-Kartendaten in 3D darzustellen. Dabei werden Kartentiles auf einen Globus projiziert und zusätzlich Gebäude eingeblendet, welche in den OSM-Daten repräsentiert sind. Das Projekt wurde an der Universität Heidelberg entwickelt und wird weiterhin von dieser gewartet. In einer Beispielanwendung wird das Java-Applet "XNavigator"benutzt, um die 3D-Daten zu visualisieren [Uni13].

Funktionsumfang und Darstellung

OpenStreetMap-3D unterscheidet sich stark von anderen Renderern, da es nicht im eigentlichen Sinn Karten rendert, sondern existierende Karten nimmt und diese als Textur für ein 3D-Modell benutzt. So können verschiedene Karten als Ausgangsmaterial genutzt werden, wodurch man selbst bestimmt, wie die endgültige Karte aussieht. Zusätzlich können nicht nur Gebäude und POIs, sondern auch Wege eingeblendet werden. Zudem werden Höhendaten genutzt, um das Relief der Erde nachzustellen. Die Anwendung auf der Website läuft bedingt flüssig und fehlerfrei. Einige der angebotenen Funktionen funktionieren nicht, zudem stockt die Anwendung oft und hat viele Darstellungsfehler.

Installation und Voraussetzungen

Der Endbenutzer benötigt, in der Beispielimplementierung, Java, um das Java-Applet ausführen zu können. Zusätzlich wird eine gute Internetverbindung empfohlen, da sehr große Datenmengen zum Anzeigen des 3D-Modells gesendet werden.

Um die Kartendaten zu erstellen, werden viele aufwendige Rechenoperationen durchgeführt, dementsprechend wird ein leistungsfähiger Server benötigt. Zudem sind die Kartendaten sehr groß, für die ganze Welt werden in etwa 500 GB Speicher benötigt.

Lizenzen

Es gibt keine klaren Angaben, ob und wenn ja unter welcher Lizenz OpenStreetMap-3D steht. Der Quellcode steht nicht offen zur Verfügung. Die Beispielimplementierung auf der Website kann kostenlos benutzt werden, dient allerdings nur als Demonstration. Es wird darauf hingewiesen, dass man sich bei Interesse direkt an die Verwaltung von OpenStreetMap-3D wenden soll.

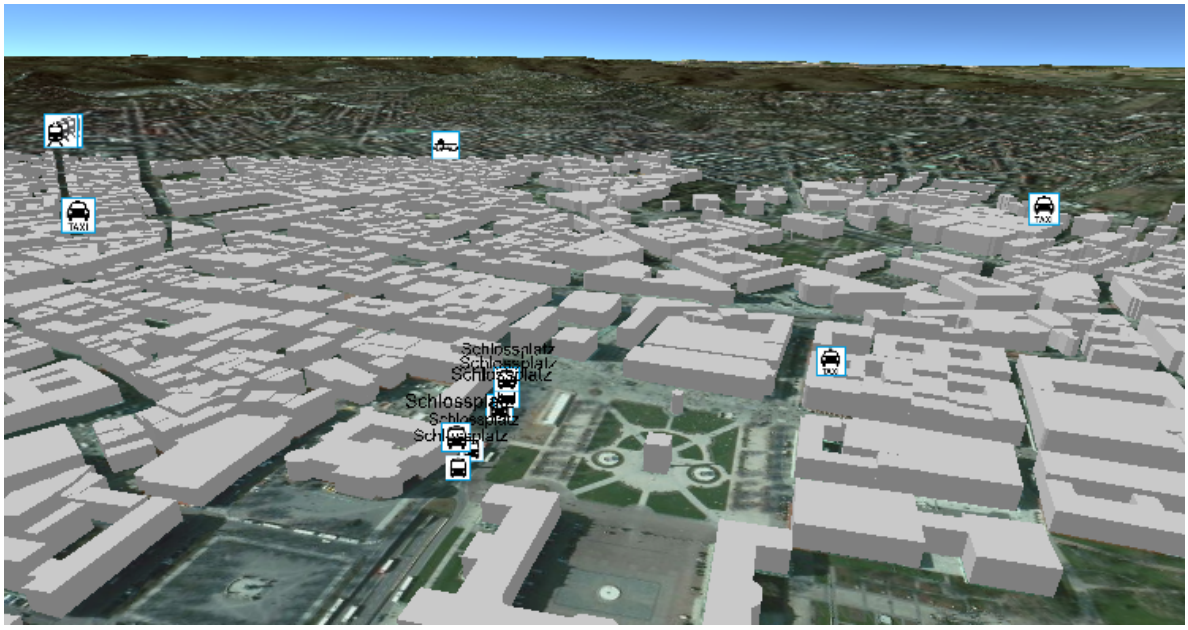

Abbildung 4.26: Stuttgart gerendert mit OpenStreetMap-3D - Darstellung mit Satellitenfotos von Bing

4.3.5 OpenStreetPad

Allgemeine Übersicht

OpenStreetPad ist ein Ein-Mann Projekt und hauptsächlich deswegen nennenswert, weil es ein Kartenrenderer für iPads ist [Ope13b]. Das Projekt ist nicht sehr ausgereift und enthält viele Fehler, zudem hat der Autor angegeben, dass er es nicht weiterentwickeln kann. Dennoch können zumindest einzelne Kartenausschnitte dargestellt werden. Das Projekt steht auf GitHub offen zur Verfügung und darf weiterentwickelt werden [Ope13c].

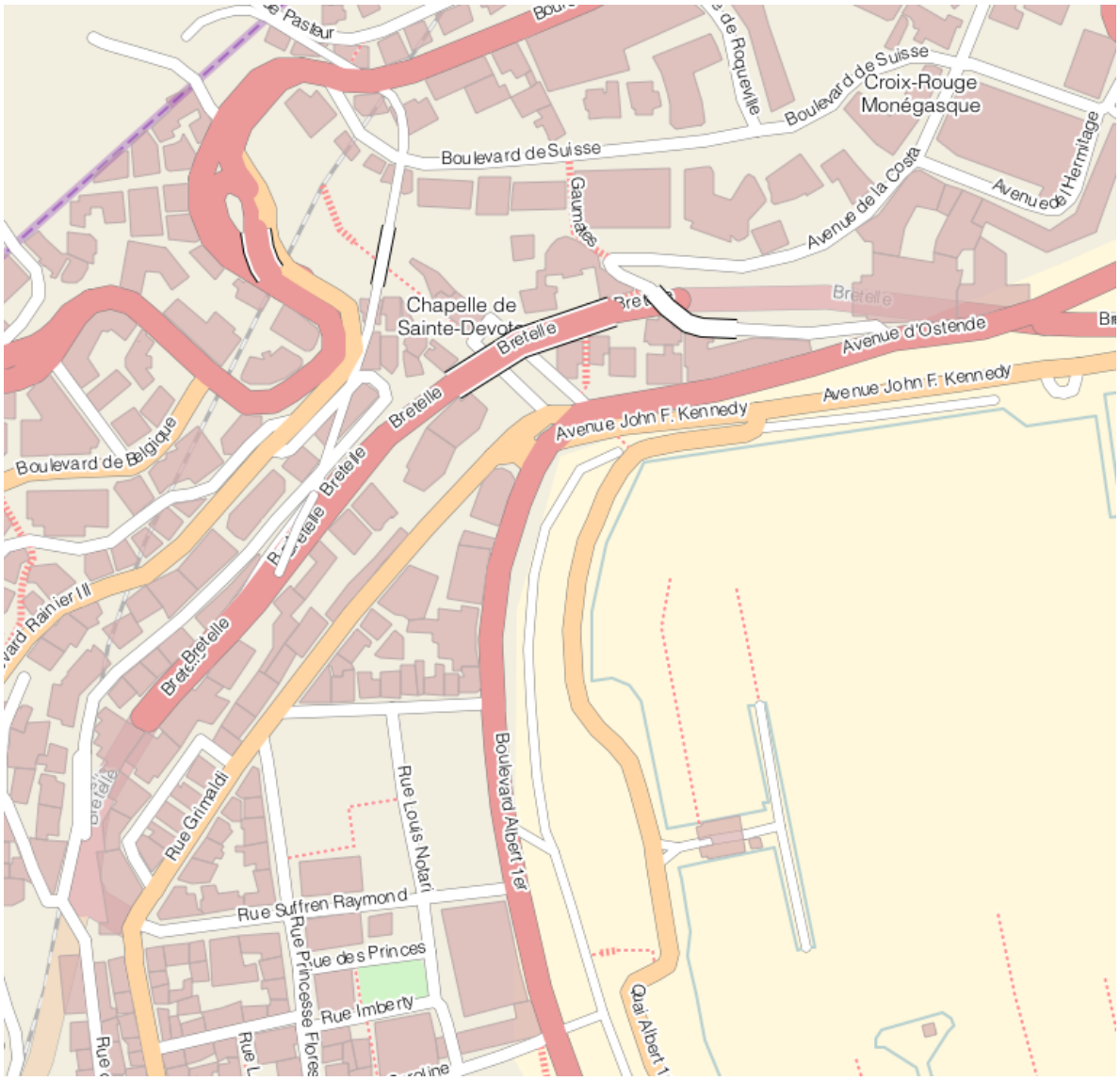

Abbildung 4.27: Monaco gerendert mit OpenStreetPad - enthält viele Darstellungsfehler

4.3.6 OSM2World

Allgemeine Übersicht

OSM2World ist ein Java-Programm, das OSM-Daten in 3D darstellen kann. Durch eine Konfigurationsdatei lassen sich diverse Einstellungen ändern. Außerdem ist es hier möglich, eigene Texturen und 3D-Modelle für die Darstellung anzugeben.

Funktionsumfang und Darstellung

Da das Programm offline läuft, benötigt OSM2World eine OSM-Datei als Eingabe. Ein automatischer Kartendownload ist nicht möglich. Diese Datei wird verarbeitet (was für ein kleines Dorf etwa eine Minute in Anspruch nimmt) und dann dargestellt. Sind keine Attribute für die Häuser angegeben, erstellt OSM2World eigene Modelle anhand der Grundfläche der Häuser. Als Dach wird dann stets ein Flachdach verwendet. Auch für Bäume und Schilder sind bereits Standard-Modelle vorhanden. Für die Höhenberechnung bietet das Programm verschiedene Verfahren an. Einerseits können Höhendaten anhand verschiedener Tags berechnet werden, andererseits können auch direkt Höhendaten verarbeitet werden. Die Steuerung erfolgt entweder durch Tastatur-Eingaben oder durch die üblichen Mausgesten. Es kann sowohl die Kamera, als auch die Position in allen Dimensionen frei bewegt werden. OSM2World unterstützt bisher kein Labeling. Man merkt dem Programm an diversen Stellen an, dass es noch in einer sehr frühen Entwicklungsphase steckt. Einerseits stürzte das Programm bei diversen Versuchen ab, andererseits wird das Programm bereits bei der Darstellung kleinerer Städte so langsam, dass es nicht mehr zu benutzen ist. Dasselbe Phänomen tritt auch auf, wenn man den Faktor für die Baumdichte auf einen realistischen Wert setzt. Nur wenn ein Wald nur noch aus vereinzelten Bäumen besteht, ist die Rendergeschwindigkeit annehmbar.

Installation und Voraussetzungen

Die Einrichtung des Programms ist simpel. Es muss ein Archiv heruntergeladen und entpackt werden [OSM13a]. Für Linux und Windows stehen Batchdateien zur Verfügung, mit der das Programm einfach gestartet werden kann. Dazu muss jedoch die JRE auf dem System laufen. Außerdem benötigt das Programm bis zu 2GB Arbeitsspeicher. Ein entsprechend gut ausgerüsteter Rechner wird deshalb von uns empfohlen.

Lizenzen

OSM2World ist Open Source und wird unter der LGPL-Lizenz angeboten. Die Benutzung ist sowohl für die private als auch für die kommerzielle Anwendung kostenfrei [OSM13b].

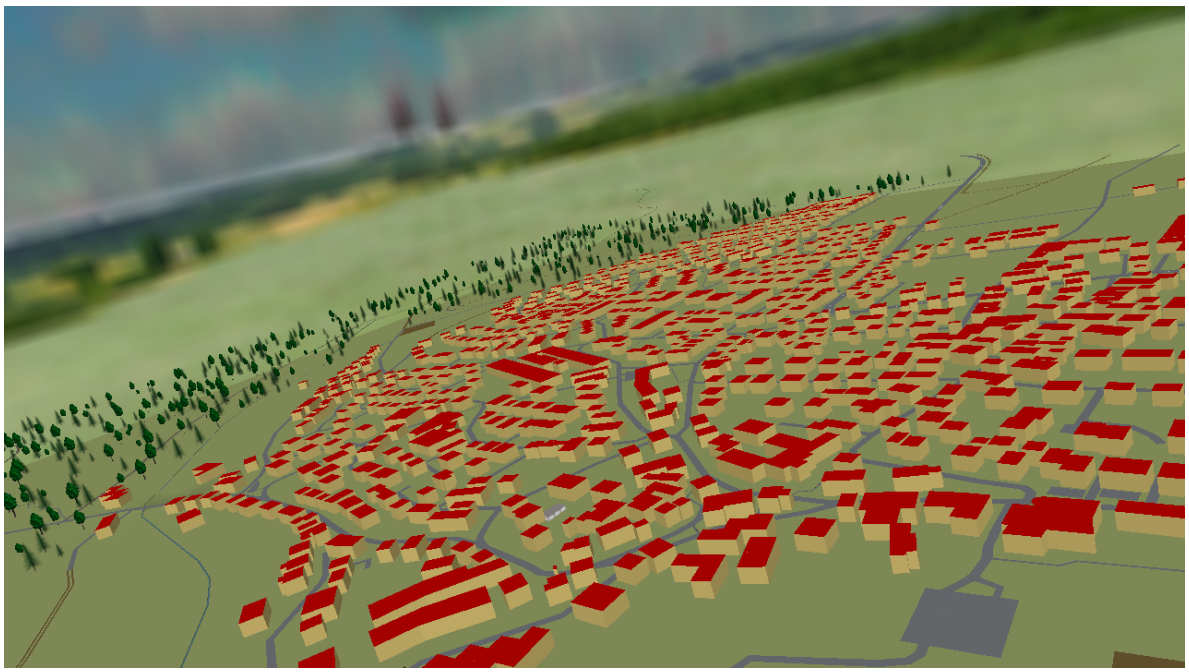

Abbildung 4.28: Musberg gerendert von OSM2World - Größere Städte überfordern OSM2World

4.3.7 Pyrender

Allgemeine Übersicht

Pyrender ist ein Python Programm, dass nicht mehr weiterentwickelt wird und von dem auch keine stabile lauffähige Version vorhanden ist [Ope13q]. Pyrender kann Karten erstellen, sowie einen Tileserver betreiben.

Funktionsumfang und Darstellung

Pyrender ist in der Lage, das Kartenmaterial automatisch zu rendern und als PNG oder Tileserver zur Verfügung zu stellen. Das OSM-Kartenmaterial wird automatisch heruntergeladen. Pyrender besitzt eine grafische Oberfläche auf HTML-Basis, um einen interaktiven Renderausschnitt anzuzeigen. Über Regeln kann der Style der erzeugten Karte angepasst werden. Es können keine Höhendaten verarbeitet werden.

Installation und Voraussetzungen

Pyrender benötigt Python Imaging Library und Cairo. Die einzelnen Komponenten (Renderer, Tile-Server usw.) müssen nacheinander manuell gestartet werden.

Lizenzen

Die Nutzung von Pyrender ist kostenlos. Der Source Code wurde unter GNU V3 veröffentlicht.

4.3.8 RenderOSM

Allgemeine Übersicht

RenderOSM wurde an der Universität Stuttgart als Studentenprojekt mit dem Ziel entwickelt, einen parametrierbaren Echtzeit-Renderer für OSM-Daten zu haben. Es gibt zwei Implementierungen, eine für Android-Smartphones und eine für moderne Browser. Beide nutzen die Grafikkarte um zu rendern, dafür werden OpenGL und WebGL verwendet.

Funktionsumfang und Darstellung

Das Projekt versucht, die gesamte Datenverarbeitung selbst zu lösen. Zu diesem Zweck wurde ein Importer für OSM-Daten geschrieben, welcher diese filtert und in eine PostgreSQL Datenbank speichert. Mithilfe eines Servers werden die Daten an die beiden Client-Typen übertragen, welche diese verarbeiten und anschließend darstellen. Da RenderOSM keine Kartentiles erzeugt, sondern Wege etc. in Echtzeit rendert, wird die Darstellung der Karten ebenfalls von RenderOSM durchgeführt. RenderOSM ist zwar lauffähig, aber nicht sehr robust und unterstützt zudem nicht alle Funktionen, die man aus modernen Kartenanwendungen gewohnt ist. Es ist möglich, POIs darzustellen und Routen einzublenden, außerdem existiert ein einfacher Straßen-Label-Algorithmus. Hauptproblem ist das bisher ungelöste Probleme mit der Speicherverwaltung. Da sehr große Datenmengen verarbeitet werden, kommt es zu erheblichen Performance-Einbußen.

Installation und Voraussetzungen

Die Installation ist relativ aufwendig, da viele Einzelschritte durchgeführt werden müssen. Es existiert eine Anleitung, welche den Idealverlauf einer Installation beschreibt, die allerdings keine weiterführende Hilfe bietet. Es wird ein leistungsfähiger Server benötigt, auf dem eine JavaVM läuft und PostgreSQL 9.1 mit der Erweiterung PostGIS 1.5 installiert ist. Da für einen Kartenausschnitt mit unter 10MB an Daten transferiert werden müssen, empfiehlt sich die Ausführung in einem Intranet. Der Endanwender benötigt entweder einen modernen Browser mit JavaScript und WebGL-Unterstützung oder ein Android-Smartphone mit Android-Version 2.2 oder höher.

Lizenzen

RenderOSM steht unter der Apache License, Version 2.0. Der Code steht jedoch nicht offen zur Verfügung. Bei Interesse muss man sich direkt an die Universität Stuttgart wenden.

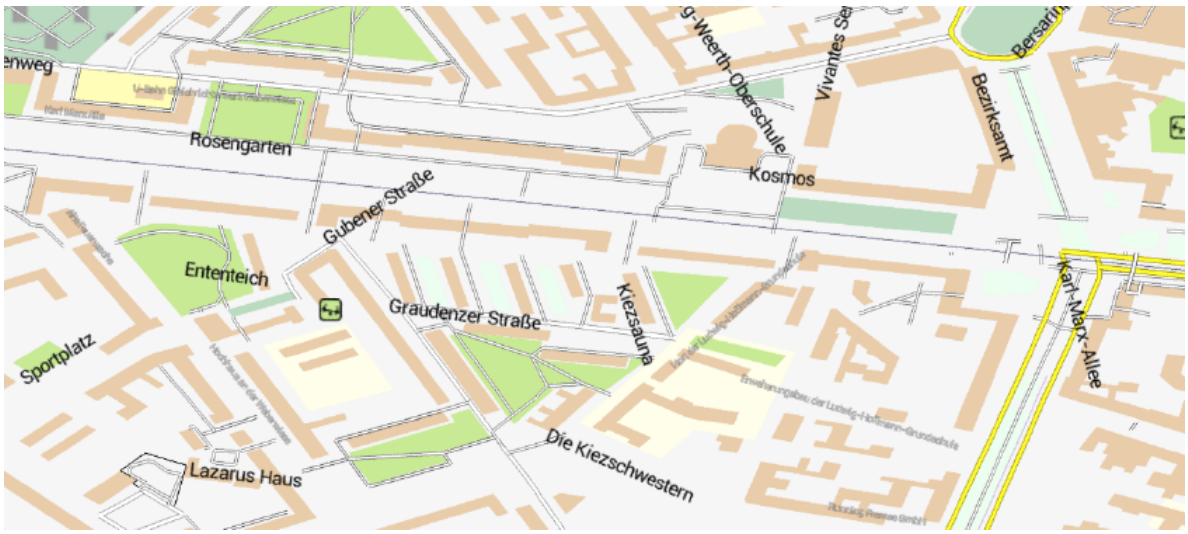

Abbildung 4.29: Berlin gerendert mit RenderOSM - Labels oft fehlerhaft

5 Bewertung der Kartenrenderer

5.1 Usecases

Zur Bewertung der untersuchten Kartenrenderer werden die folgenden Usecases herangezogen. Die Usecases sollen ein breites Nutzungsspektrum abbilden. Die drei eingesetzten Usecases bilden einen Großteil des Nutzungsrahmens für Kartenrenderer ab. Jeder Usecase steht repräsentativ für eine Klasse von ähnlichen Anwendungsfällen. Bei der Beurteilung der Eignung eines Kartenrenderer zur Umsetzung eines Usecases werden die Anforderungen mit den Eigenschaften des untersuchten Kartenrenderer abgeglichen, sowie eine Einschätzung vorgenommen.

5.1.1 Printprodukte

Das Veranstaltungbsüro eines Festivals möchte einen Übersichtsflyer erstellen. Dieser soll eine Karte mit den einzelnen Veranstaltungsorten enthalten, die über eine Stadt verteilt sind. Die Karte wird von einem Mitarbeiter der Grafikdesign-Abteilung erstellt.

Anforderungen

- Einfache Bedienung
- Externe POIs mit Icon einfügen
- Style der Karte anpassen
- Druckbares Format (z.B. Vektorgraphik)

5.1.2 Mobile Applikationen

Für eine Smartphone-App wird eine interaktive Karte benötigt, auf der die aktuelle Position sowie weitere Informationen dargestellt werden können. Je nach Bedarf soll die Kartenanwendung erweiterbar sein.

Anforderungen

- API für iOS, Android und evtl. Windows Phone 8
- Externe POIs mit Icon einfügen
- Style der Karte anpassen

5.1.3 Webanwendung

Für eine Vereinshomepage benötigt ein ehrenamtlicher Webdesigner eine Karte um den Standort des Vereinsheims zu markieren. Die Karte soll in einem Format vorliegen, dass in eine Homepage eingebunden werden kann. Zudem sollte die Nutzung des Kartenmaterials kostenlos sein.

Anforderungen

- Verständliche Dokumentation und Einarbeitung
- Einfügen eines POIs oder Nachbearbeitung möglich
- Kostenlose Nutzung für Vereinshomepage
- Export in Webformate, bzw. Einbindung in Homepage möglich.

5.2 Bewertung

Bei der Beurteilung wurden die kommerziellen Produkte nachrangig behandelt, da diese durch Lizenzgebühren und unveröffentlichten Quellcode in ihrer Weiterentwicklung und Analyse seitens der Nutzer eingeschränkt sind.

5.2.1 Usecase: Printprodukte

Folgende Kartenrenderer eignen sich zur Umsetzung des Usecases Printprodukte.

Vorschlag 1: Tile Mill

Die Kartenerstellung via TileMill ist dank dem GUI und der ausführlichen Erläuterungen des Herstellers ohne großen Administrationsaufwand seitens der Einrichtung durchzuführen. Die Anzeige der Karten kann komplett mit CartoCCS Styles angepasst werden. Auch große Datensätze oder einzelne POIs können auf der Karten angezeigt werden. Die Einbindung einzelner Positionsdaten ist allerdings aufwendiger als mit den populären Browser-Tools wie Google Maps.

Vorschlag 2: MapOSMatic

Die Kartenerstellung via Browser ist sehr einfach und ohne großen Administrationsaufwand seitens der Einrichtung durchzuführen. Die fehlende Parametrierung der Karte wird durch die verschiedenen Stile kompensiert. Der Export als SVGZ Grafik eignet sich gut zur Erstellung von Printprodukten. Allerdings besteht keine Möglichkeit die Grafik mit Overlays zu Render. Dies schränkt die Nutzung für Anwendungsfälle mit vielen verteilten Datenpunkten ein.

5.2.2 Usecase: Mobile Applikationen

Folgende Kartenrenderer eignen sich zur Umsetzung des Usecases Mobile Applikationen.

Vorschlag 1: MapsWithMe

Der Kartenrenderer MapsWithMe ist speziell für die Erstellung von mobilen Anwendungen zugeschnitten. Zu beachten ist, dass Karten nur offline genutzt werden können, was jedoch bei bestimmten Anwendungsfällen, wie zum Beispiel Reiseführern, besonders nützlich sein kann. Es ist möglich, POIs zu markieren, allerdings können Karten nicht weiter angepasst werden. Zudem steht nur die API zur Verfügung, es ist nicht möglich, die Anwendung selber zu erweitern.

Vorschlag 2: Mapsforge

Für bestimmte Anwendungsfälle kann Mapsforge eine gute Alternative sein. Die API kann einfach eingebunden werden und darf beliebig erweitert werden. Zudem können Karten nach eigenen Wünschen verändert können. Wenn man zum Beispiel einen kleineren Kartenausschnitt in einem sehr ungewöhnlichen Layout offline rendern möchte, stellt Mapsforge hierfür die einzige uns bekannte Möglichkeit dar.

5.2.3 Usecase: Webanwendung

Folgende Kartenrenderer eignen sich zur Umsetzung des Usecases Webanwendung.

Vorschlag 1: Open Street Browser

Die Karten von Open Street Browser sind leicht und schnell zu exportieren. Falls der gewünschte POI noch nicht verfügbar ist, muss dieser mit einem weiterem Tool auf der exportierten Karte markiert werden. Die Einbindung des Grafik ExportsAl in die Homepage erfolgt über die eingesetzte Technologie der Homepage.

Vorschlag 2: Google Maps Karte

Die Einbindung einer interaktiven Google Maps Karte bietet viele Vorteile. Neben der Markierung des POIs kann dem Besucher der Website eine Routenberechnung von einem frei wählbaren Startpunkt angegeben werden. Die Einbindung der Karte ist dank vorgeneriertem Code einfach umzusetzen.

6 Fazit und Übersicht

Dieses Kapitel bietet eine Übersicht der betrachteten Renderer aus 4 Bewertung der Kartenrenderer. Diese Übersicht stellt die Ergebnisse der Bewertung mit den Prüfaspekten (siehe 2) in einer Matrix dar.

Abschließend wird noch ein Fazit über die Fachstudie gezogen.

6.1 Fazit

Nachdem wir alle Renderer analysiert und verglichen haben, sind wir in der Lage, Empfehlungen auszusprechen und Auffälligkeiten zu erläutern.

Zuerst möchten wir festhalten, dass die Auswahl an Renderern unerwartet groß ist. Für nahezu jeden denkbaren Einsatzzweck gibt es mindestens ein Programm oder einen Dienst. Gleichzeitig überschneiden sich die Fähigkeiten der Renderer oft. Es gibt beispielsweise gleich mehrere Renderer, die es sich zur Aufgabe gemacht haben, OSM-Daten in 3D darzustellen. Auch kommandozeilenbasierende, parametrierbare Renderer gibt es mehrere, die sich in den Funktionen jedoch kaum voneinander unterscheiden.

Die Stabilität und Einsetzbarkeit variiert jedoch stark von Renderer zu Renderer. Auch deshalb ist unsere Matrix, die diese Fachstudie übersichtlich zusammenfasst, sehr nützlich. In ihr kann man nämlich auf einfache Weise einen geeigneten, stabil laufenden Renderer für die jeweiligen Ansprüche finden. Gleichzeitig sieht man Alternativen und deren Unterschiede.

Natürlich gibt es auch Programme oder Dienste, die gut funktionieren und von uns deshalb eine Empfehlung erhalten. Empfehlungen für spezielle Anwendungsszenarien können im Kapitel Usecases gefunden werden. An dieser Stelle möchten wir ganz allgemein verschiedene Projekte empfehlen.

Da wir während der Recherchen oft Probleme damit hatten, die Programme zum Laufen zu bringen möchten wir bei unseren Empfehlungen die Programme bevorzugen, die einfach einzurichten sind. Vor allem Open Source-Programme sind (besonders unter Linux) meist sehr schwer einzurichten. Erschwerend kommt hinzu, dass die Dokumentation bzw. Installationsanweisungen oft sehr dürftig ausfallen.

Deshalb möchten wir zunächst GoogleMaps empfehlen. GoogleMaps bietet sehr viele Funktionen, die ausgesprochen gut funktionieren. Damit ist es vielen anderen Diensten und Programmen sehr weit voraus. Gleichzeitig ist GoogleMaps einfach zu benutzen und läuft auf einer Vielzahl von Betriebssystemen. So werden die meisten Browser und nahezu alle Smartphone-Betriebssysteme unterstützt (siehe auch Matrix). Für kleinere Anwendungen ist es zudem kostenlos.

Jedoch möchten wir die vielen Open Source-Programme natürlich nicht vollkommen außer Acht lassen. Gerade wenn die Darstellung möglichst detailliert bearbeitet werden soll, ist man bei den Open Source-Programmen in vielen Fällen gut aufgehoben. Zwar gibt es einige, die sich noch in der Entwicklung befinden, oder nie fertig entwickelt und aufgegeben wurden, jedoch gibt es auch einige ausgesprochen gute Exemplare.

Eines davon ist Mapnik. Mapnik ist ein API, das das Rendern statischer Karten erlaubt und für viele verschiedene Programmiersprachen angeboten wird. Zusätzlich ist es unter iOS einsetzbar. Mapnik bietet sehr viele Möglichkeiten, die Kartendarstellung zu parametrieren und bietet ein hervorragendes Labeling. Gleichzeitig ist Mapnik sehr stabil und wird deshalb auch von vielen Projekten eingesetzt. Aufgrund dieser Tatsachen können wir Mapnik empfehlen.

Wer bereit ist, Geld auszugeben, sich dafür aber um nichts kümmern zu müssen, ist mit OpenCycleMap sehr gut beraten. Dieses bietet eine Leaflet-konforme Schnittstelle an und ist damit sehr einfach in Projekte einzubinden. OpenCycleMap erstellt auf Wunsch und gegen Bezahlung Karten, die nach eigenen Wünschen gestaltet werden können. Diese werden dann auf den Servern von OpenCycleMap gehostet und ständig aktuell gehalten, was sehr komfortabel ist. Deshalb möchten wir auch OpenCycleMap empfehlen.

Weiterhin möchten wir noch einen weiteren Renderer hervorheben. MapsWithMe ist eines der wenigen Programmen, das Karten auf Smartphones offline rendern kann. Außerdem bietet MapsWithMe ein API an, um in eigene Apps integriert werden zu können. Damit eignet es sich ideal dazu, Apps zu entwickeln, die auch ohne Internetverbindung funktionieren.

Abschließend möchten wir noch darauf hinweisen, dass es aus unserer Sicht aktuell nicht mehr notwendig ist, ein weiteres Projekt zum Rendern von Karten zu beginnen. Jede Funktion, die sinnvoll ist, gibt es momentan in einem oder mehreren Projekten. Wir hatten bei unseren Recherchen das Gefühl, dass sich einige Projekte zuvor nicht erkundigt haben, was es aktuell an entsprechender Software gibt. Denn viele Projekte wurden vermutlich auch deshalb wieder eingestellt, da eingesehen wurde, dass ein identisches, oder ähnliches Projekt bereits existiert. Wir hoffen deshalb, dass wir mit dieser Fachstudie dazu beitragen können, eine bessere Übersicht über die verfügbaren Renderer zu geben.

6.2 Übersicht

		Finanzen			Funktionsumfang										Darstellung					Voraussetzungen			Installation					
		Lizenz	OpenSource	Kosten	Slippy Map	Offline	API	GUI	Parametrierung	Kartendownload	Höhendaten	Overlay	Kartenquelle	Grafikkartennutzng	Routing	Subj. Eindruck	Performance	Labeling	3D	Ausgabe	Datenbank	Server	Plattform	Sprache	Kompiliert	Installer	Dauer	Weiterentwicklung	Einsetzbar
komerziell	ADAC Maps	[10]	✗	0-15	✓	✗	✗	✓	✗	✓	✓	✓	P,NT,A	✗	✓	4	4	✓	✗	✗	✗	✗	B	JS	✓	✓	✓	✓	✓
	Apple Maps	[10]	✗	0	✓	✗	✗	✓	✗	✓	✓	✓	AP	✗	✓	5	5	✓	✓	✗	✗	✗	I	O	✓	✓	2	✓	✓
	Bing Maps	[10]	✗	0-?	✓	✗	✗	✓	✗	✓	✓	✓	MS,N	✗	✓	5	5	✓	✗	✗	✗	✗	B	H5	✓	✓	✓	✓	✓
	CartoType	[10]	✗	3000	✗	✓	✓	✓	✓	✗	✓	✓	O,E	✗	✓	3	4	✓	✗	✓✗	✗	✗	M,A	C++,C,J,O	✓	?	?	✓	✓
	Google Maps	[10]	✗	0-?	✓	✓✗	✓	✓	✗	✓	✓	✓	G	✓	✓	5	5	✓	✓	✗	✗	✗	M,S	HR, JS	✓	✓	✓	✓	✓
	TileMill	[10]	✗	0-500	✗	✓	✓	✓	✓	✓	✓	✓	O	✗	✗	4	4	✓	✗	S,P,N	✓✗	✗	M	✓	✓	✓	3	✓	✓
	Mappy	[10]	✗	0-?	✓	✗	✓	✓	✗	✓	✗	✓	A,TA	✗	✓	4	5	✓	✗	✗	✗	✗	B,I,A	JS	✓	✓	✓	✓	✓
	MapsWithMe	[10]	✗	0-4	✓	✓	✓	✓	✗	✓	✗	✓	O	?	✗	5	5	✓	✗	✗	✗	✗	A,I	J,O	✓	✓	2	✓	✓
	Nokia Maps	[10]	✗	0-1500	✓	✓	✓	✓	✗	✓	✓	✓	N	✗	✓	5	5	✓	✓	✗	✗	✗	B,P,A	H5	✓	✓	✓	✓	✓
	TomTom	[10]	✗	?	✓	✗	✓	✓	✗	✓	✗	✓	?	✗	✓	4	3	✓	✗	✗	✗	✗	B,S	JS,J,O	✓	✓	2	✓	✓
Freeware, lauffähig	Ceyx	[5]	✓	0	✗	✓	✗	✗	✓	✗	✗	✗	O	✗	✗	3	4	✓	✗	N	✗	✗	M	Py	✗	✗	20	✓	✓
	Kendzi3D	?	✓	0	✓	✓	✗	✓	✗	✓	✗	✗	O	✓	✗	2	2	✗	✓	✗	✗	✗	M	J	✓	✓	5	✓	✓
	KothicJS	[3]	✓	0	✗	✗	✓	✗	✓	✗	✗	✓	O	✗	✗	3	2	✓	✗	✗	✗	✗	B	JS	✓	✓	✓	✓	✓
	Maperative	[10]	✗	0	✓	✓	✓	✓	✓	✓	✓	✓	O	✗	✗	3	3	✓	✓	S,B,C	✗	✗	M	Py	✓	✓	4	✓	✓
	Mapnik	[7]	✓	0	✗	✓	✓	✗	✓	✗	✗	✓	O	✗	✗	4	5	✓	✗	P,S,N,J	✓✗	✗	W,L,I	Py,CP	✗	✗	30	✓	✓
	MapOSMatic	[1]	✓	0	✗	✓	✗	✓	✗	✓	✗	✗	O	✗	✗	4	4	✓	✗	S,P,N	✗	✗	B	JS	✓	✓	✓	✓	✓
	Mapsforge	[7]	✓	0	✓	✓	✓	✗	✓	✓	✗	✓	O	✓	✗	4	2	✓	✗	✗	✗	✗	A	J	✓	✓	✓	✗	✓
	Mapweaver	?	✓	0	✗	✓	✗	✗	✓	✓	✓	✓	O	✗	✗	4	4	✓	✗	S,P	✗	✗	L	Pl	✗	✗	15	✓	✓
	Memphis	[8]	✓	0	✗	✓	✓	✗	✓	✗	✗	✗	O	✗	✗	1	3	✗	✗	N	✗	✗	L	C, GO	✗	✗	30	✗	✓
	OpenCycleMap	[10]	✗	0-300	✓	✗	✓	✓	✗	✓	✓	✗	O	✗	✗	4	5	✓	✗	✗	✗	✗	B	JS	✓	✓	✓	✓	✓
	OpenStreetBrowser	[1]	✓	0	✓	✗	✓	✓	✗	✓	✗	✓	O	✗	✓	4	4	✓	✗	✗	✗	✗	B	JS	✓	✓	✓	✓	✓
	Osmand	[10]	✓	0-6	✓	✓	✗	✓	✗	✓	✗	✓	O	?	✓	4	4	✓	✗	✗	✗	✗	A	J	✓	✓	1	✓	✓
	Osmarender	[4]	✓	0	✗	✓	✗	✗✓	✓	✗	✗	✓	O	?	✗	5	2	✓	✗	S	✗	✗	M	X	✓	✓	15	✗	✓
	SMRender	[6]	✓	0	✗	✓	✗	✗	✓	✗	✗	✗	O	✗	✗	3	4	✓	✗	P,N	✗	✗	L,W	C	✗	✗	20	✓	✓
nicht lauffähig	Cartagen	[9]	✓	0	✓	✗	✓	✓	✓	✓	✗	✓	O	✗	✗	1	1	✓	✗	✗	✗	✗	B	JS	✓	✓	✓	✗	✗
	Kosmos	[3]	✓	0	✓	✓	✗	✓	✓	✓	✓	✗	O	✗	✗	3	✓	✓	✗	N,B	✗	✓✗	W	C#	✓	✗	4	✗	✗
	Kothic	?	✓	0	✓	✓	✗	✓	✓	✗	✗	✓	O	✗	✗	3	2	✓	✗	✗	✓	✗	L	Py	✗	✗	30	✓	✗
	OpenStreetMap-3D	[4]	✓	0	✓	✗	✗	✓	✓✗	✓	✓	✓	Alles	✓	✓	3	2	✓✗	✓	✗	✓✗	✓✗	B	J	✓	✓	✓	✗	✗
	OpenStreetPad	[3]	✓	0	✓	✗	✗	✓	✓	✓	✗	✗	O	✗	✗	2	0	✗	✓	✗	✗	✗	I	O	✓	✓	2	✗	✗
	OSM2World	[7]	✓	0	✓	✓	✗	✓	✓	✗	✓	✗	O	✓	✗	2	1	✗	✓	✗	✗	✗	M	J	✓	✗	5	✓	✗
	Pyrender	[6]	✓	0	✓	✗	✗	✗	✓	✓	✗	✗	O	✗	✗	0	0	✗	✗	N	✗	✓	L	Py	✗	✗	30	✗	✗
	RenderOSM	[2]	✓	0	✓	✗	✓	✓	✓	✓	✓	✓	O	✓	✗	3	3	✓	✗	✗	✓	✓	B,A	J,JS	✓	✓	2	✗	✗

Tabelle 6.1: Übersicht Renderer:

Lizenzen: [1]: AGPL v3, [2]: Apache Licence v2, [3]: BSD, [4]: GPL v2, [5]: GPL v2+, [6]: GPL v3, [7]: LGPL, [8]: LGPL v2.1+, [9]: MIT, [10]: Own.
Kartenquellen: O: Open Street Map, AP: Apple, MS: Microsoft, N: Nokia, E: ESRI, G: Google, NT: Navteq, TA: Tele Atlas, P: Public Transport Victoria, A: AND.
Ausgabe: P: PDF, S: SVG, N: PNG, C: Collada, B: Bitmap, J: JPG.
Plattform: M: Multiplattform (L, O, W), W: Windows, L: Linux, A: Android, O: Mac, S: Alle Smartphonesysteme (I,A,P), B: Browser, I: iOS, P: Windows Phone.
Sprache: JS: JavaScript, J: Java, C++: C++, C: C, Pl: Perl, Py: Python, O: Objective-C, H5: HTML-5, HR: HTML Anfragen, GO: GObject, C#: C#. X: Extensible Stylesheet Language (XSL)

6.2.1 Lizenzen

Eine Lizenz legt fest, was mit einer Software gemacht werden darf. Beispielsweise ob es erlaubt ist, die Software zu verändern, weiter zu geben, oder grundsätzlich zu nutzen. Die Renderer, die wir in dieser Fachstudie untersucht haben, haben sehr unterschiedliche Lizenzen. Die meisten kommerziellen Renderer haben jeweils eine eigene Lizenz, die genau auf diese Software zugeschnitten ist. Die anderen, nicht kommerziellen Renderer nutzen meist bekannte Lizenzen. Diese möchten wir im Folgenden sehr knapp erläutern. Wir möchten darauf hinweisen, dass dies keinesfalls vollständig ist. Es geht nur darum, eine grobe Vorstellung der Lizenz zu erhalten.

AGPL

Kompatibel zu GPL. Ist fast vollständig identisch, bis auf die Tatsache, dass der Quellcode mitgeliefert werden muss – auch für die Softwarekomponenten, die unter GPL stehen. Quellcode muss insbesondere auch bei Webapplikationen (die man eigentlich nur „nutzt“ – es findet keine Auslieferung statt) mitgeliefert werden.

Apache Licence v2

Kompatibel zu GPL v3. Die Software darf überall frei benutzt, kopiert und verändert werden. Software, die Komponenten benutzt, die unter der Apache-Lizenz stehen, muss selbst nicht unter Apache-Lizenz gestellt werden. Wird Code verändert, muss dies deutlich sichtbar gemacht werden. Der Lizenztext muss immer beigefügt sein.

BSD

Die Software darf kostenfrei genutzt, verändert und weiterverteilt werden. Es ist auch erlaubt, Software, die Komponenten verwendet, die unter BSD-Lizenz stehen, zu verkaufen. Der Copyright-Vermerk darf nicht entfernt werden. BSD beinhaltet kein Copyleft – d.h. wird das ursprüngliche Programm verändert und in binärer Form, d.h. in einem nicht mehr menschenlesbaren Format, weiterverbreitet, muss der Quellcode nicht mitgeliefert werden. Die ursprüngliche Lizenz muss jedoch immer beiliegen.

GPL

Ähnlich wie BSD. Jedoch enthält die GPL Copyleft. D.h. abgeleitete Software muss unter derselben Lizenz wie die ursprüngliche Software stehen. Der Quellcode muss immer mitgeliefert werden. Die Unterschiede zwischen v2 und v3 sind relativ gering und kaum relevant für die hier behandelten Programme.

LGPL

Sehr ähnlich wie GPL, jedoch mit schwächerem Copyleft. Der Quellcode eigener Softwarekomponenten muss nicht offen gelegt werden.

MIT

Besagt, dass die Ausführung, Änderung, Fusionierung, sowie das Kopieren und der Verkauf der Software erlaubt ist, solange der Lizenz-Text beigelegt wird. Gleichzeitig schließt die Lizenz jegliche Haftung der Autoren der Software aus.

Literaturverzeichnis

[ADA13a] ADAC e.V. *ADACMaps: About*, 2013. http://maps.adac.de/Pages/about.html. (Zitiert auf Seite 21)

[ADA13b] ADAC e.V. *ADACMaps: Karte*, 2013. http://maps.adac.de/. (Zitiert auf Seite 21)

[App13a] Apple. *Apple: Framework*, 2013. http://developer.apple.com/library/ios/#documentation/MapKit/Reference/MapKit_Framework_Reference/_index.html. (Zitiert auf Seite 23)

[App13b] Apple Inc., http://gspsa21.ls.apple.com/html/attribution.html. *Kartenmaterial: Apple*, 2013. (Zitiert auf Seite 17)

[Bin13a] Bing. *Bing: API*, 2013. http://www.microsoft.com/maps/choose-your-bing-maps-API.aspx. (Zitiert auf Seite 25)

[Bin13b] Bing. *Bing: Main*, 2013. http://www.microsoft.com/maps/. (Zitiert auf Seite 25)

[Car13a] CartoType. *CartoType: Anzeige der Routenberechnung*, 2013. http://www.cartotype.com/route-finding.html. (Zitiert auf Seite 27)

[Car13b] CartoType. *CartoType: APIs und SDKs*, 2013. http://www.cartotype.com/documentation.html. (Zitiert auf Seite 27)

[Car13c] CartoType. *CartoType: Auslieferung des Source Code*, 2013. http://www.cartotype.com/features.html. (Zitiert auf Seite 28)

[Car13d] CartoType. *CartoType: CTM1*, 2013. http://www.cartotype.com/how-to-create-map-files-for-cartotype-ctm1-files.html. (Zitiert auf Seite 27)

[Car13e] CartoType. *CartoType: Download*, 2013. http://www.cartotype.com/downloads.html. (Zitiert auf Seite 27)

[Fre13a] Freie Universität Berlin. *mapsforge: Lizenz*, 2013. http://wiki.openstreetmap.org/wiki/Mapsforge. (Zitiert auf Seite 58)

[Fre13b] Freie Universität Berlin. *mapsforge: Source*, 2013. http://code.google.com/p/mapsforge/. (Zitiert auf Seite 58)

[Fun13] P. D. S. Funke. *Themenbeschreibung Fachstudie Nr. 177*, 2013. (Zitiert auf Seite 11)

[Goo13a] Google. *Google Maps: Directions API*, 2013. https://developers.google.com/maps/documentation/directions/. (Zitiert auf Seite 29)

[Goo13b] Google. *Google Maps: Distance Matrix API*, 2013. https://developers.google.com/maps/documentation/distancematrix/. (Zitiert auf Seite 30)

[Goo13c] Google. *Google Maps: Elevation API*, 2013. https://developers.google.com/maps/documentation/elevation/. (Zitiert auf Seite 30)

[Goo13d] Google. *Google Maps: Enterprise API*, 2013. http://www.google.com/enterprise/mapsearth/. (Zitiert auf Seite 30)

[Goo13e] Google. *Google Maps: Geocoding API*, 2013. https://developers.google.com/maps/documentation/geocoding/. (Zitiert auf Seite 30)

[Goo13f] Google. *Google Maps: Google Earth API*, 2013. https://developers.google.com/earth/. (Zitiert auf Seite 30)

[Goo13g] Google. *Google Maps: Image API*, 2013. https://developers.google.com/maps/documentation/imageapis/. (Zitiert auf Seite 29)

[Goo13h] Google. *Google Maps: JavaScript API*, 2013. https://developers.google.com/maps/documentation/javascript/tutorial. (Zitiert auf Seite 29)

[Goo13i] Google. *Google Maps: Lizenzen*, 2013. https://developers.google.com/maps/licensing. (Zitiert auf Seite 31)

[Goo13j] Google. *Google Maps: Places API*, 2013. https://developers.google.com/places/. (Zitiert auf Seite 30)

[Ken13a] Kendzi 3D. *Kendzi 3D: Roof*, 2013. http://wiki.openstreetmap.org/wiki/JOSM/Plugins/Kendzi3D/tutorial/skillion_roof. (Zitiert auf Seite 48)

[Ken13b] Kendzi 3D. *Kendzi 3D: Window*, 2013. http://wiki.openstreetmap.org/wiki/JOSM/Plugins/Kendzi3D/tutorial/windows. (Zitiert auf Seite 48)

[Kot13a] Kothic. *Kothic: Downloads*, 2013. https://github.com/kothic/kothic. (Zitiert auf Seite 78)

[Kot13b] Kothic. *KothicJS: Karte*, 2013. http://kothic.org/js/. (Zitiert auf Seite 50)

[Kot13c] Kothic. *KothicJS: Readme*, 2013. https://github.com/kothic/kothic-js#readme. (Zitiert auf Seite 50)

[Kot13d] Kothic. *KothicJS: Sourcecode*, 2013. https://github.com/kothic/kothic-js. (Zitiert auf Seite 50)

[Map13a] Mapbox. *Tilemill: Installation*, 2013. http://www.mapbox.com/tilemill/docs/win-install/. (Zitiert auf Seite 42)

[Map13b] Mapbox. *Tilemill: Kosten*, 2013. http://www.mapbox.com/plans/. (Zitiert auf Seite 42)

[Map13c] Mapbox. *Tilemill: Lizenz*, 2013. http://www.mapbox.com/tos/. (Zitiert auf Seite 42)

[Map13d] Mapbox. *Tilemill: SDKs und APIs*, 2013. http://www.mapbox.com/tour/. (Zitiert auf Seite 41)

[Map13e] Maperitive. *Maperitive: Funktionen*, 2013. http://maperitive.net/. (Zitiert auf Seite 52)

[Map13f] Maperitive. *Maperitive: Lizenz*, 2013. http://maperitive.net/docs/FAQ.html. (Zitiert auf Seite 53)

[Map13g] Maperitive. *Maperitive: Symbole*, 2013. http://maperitive.net/docs/Symbols/icon.html. (Zitiert auf Seite 52)

[Map13h] Mapnik. *Mapnik:API*, 2013. http://mapnik.org/docs/v2.0.1/api/python/. (Zitiert auf Seite 54)

[Map13i] Mapnik. *Mapnik:Output*, 2013. https://github.com/mapnik/mapnik/wiki/OutputFormats. (Zitiert auf Seite 54)

[Map13j] Mapnik. *Mapnik:Plugin*, 2013. https://github.com/mapnik/mapnik/wiki/PluginArchitecture. (Zitiert auf Seite 54)

[Map13k] Mapnik. *Mapnik:Symbology*, 2013. https://github.com/mapnik/mapnik/wiki/SymbologySupport. (Zitiert auf Seite 54)

[Map13l] MapOSMatic. *MapOSMatic: About*, 2013. http://www.maposmatic.org/about/. (Zitiert auf Seite 56)

[Map13m] MapsWithMe. *MapsWithMe:API*, 2013. https://github.com/mapswithme/api-android. (Zitiert auf Seite 35)

[Map13n] MapsWithMe. *MapsWithMe:Features*, 2013. http://mapswith.me/en/features/. (Zitiert auf Seite 35)

[Map13o] MapsWithMe. *MapsWithMe:Play*, 2013. https://play.google.com/store/apps/details?id=com.mapswithme.maps.pro. (Zitiert auf Seite 35)

[Map13p] Mapweaver. *Mapweaver: Features*, 2013. http://wiki.openstreetmap.org/wiki/Mapweaver#Outstanding_features. (Zitiert auf Seite 60)

[Map13q] Mapweaver. *Mapweaver: Installation*, 2013. http://wiki.openstreetmap.org/wiki/Mapweaver#Installation. (Zitiert auf Seite 60)

[Map13r] Mapweaver. *Mapweaver: Source*, 2013. http://wiki.openstreetmap.org/wiki/Mapweaver#Source. (Zitiert auf Seite 60)

[MIT13a] MIT. *Cartagen: Karte*, 2013. http://cartagen.org/. (Zitiert auf Seite 75)

[MIT13b] MIT. *Cartagen: OSM Wiki*, 2013. http://wiki.openstreetmap.org/wiki/DE:Cartagen. (Zitiert auf Seite 75)

[MIT13c] MIT. *Cartagen: Sourcecode*, 2013. https://github.com/jywarren/cartagen. (Zitiert auf Seite 76)

[Nok13a] Nokia. *HERE: Kosten*, 2013. http://developer.here.com/de/plans. (Zitiert auf Seite 37)

[Nok13b] Nokia. *HERE: Native APIs*, 2013. http://developer.here.com/de/native-apis. (Zitiert auf Seite 37)

[Nok13c] Nokia. *HERE: Web APIs*, 2013. http://developer.here.com/de/web-apis. (Zitiert auf Seite 37)

[Nok13d] Nokia, http://here.com/terms#mapsAdditionalTermsDiv. *Kartenmaterial: Nokia*, 2013. (Zitiert auf Seite 18)

[Ohl13] Ohloh. *Memphis: Source Code*, 2013. http://www.ohloh.net/p/libmemphis. (Zitiert auf Seite 62)

[Ope13a] Open Street Map, http://www.openstreetmap.org/copyright. *Kartenmaterial: Open Streetmap*, 2013. (Zitiert auf Seite 18)

[Ope13b] Open Street Map. *OpenStreetPad: OSM Wiki*, 2013. http://wiki.openstreetmap.org/wiki/OpenStreetPad. (Zitiert auf Seite 82)

[Ope13c] Open Street Pad SVN. *OpenStreetPad: Sourcecode*, 2013. https://github.com/beelsebob/OpenStreetPad. (Zitiert auf Seite 82)

[Ope13d] OpenStreetBrowser. *OpenStreetBrowser: Wiki*, 2013. http://wiki.openstreetmap.org/wiki/OpenStreetBrowser/Instructions. (Zitiert auf den Seiten 66 und 67)

[Ope13e] Openstreetmap. *Ceyx: Lizenz*, 2013. http://wiki.openstreetmap.org/wiki/Ceyx. (Zitiert auf Seite 47)

[Ope13f] Openstreetmap. *Ceyx: Source Code*, 2013. http://wiki.openstreetmap.org/wiki/Ceyx. (Zitiert auf Seite 47)

[Ope13g] Openstreetmap. *Kosmos: Lizenz und Source Code*, 2013. http://wiki.openstreetmap.org/wiki/Kosmos_Development. (Zitiert auf Seite 77)

[Ope13h] Openstreetmap. *Kosmos: Wiki Openstreetmap*, 2013. http://wiki.openstreetmap.org/wiki/DE:Kosmos. (Zitiert auf Seite 77)

[Ope13i] OpenStreetMap. *OpenCycleMap: API*, 2013. http://www.thunderforest.com/tutorials/. (Zitiert auf Seite 64)

[Ope13j] OpenStreetMap. *OpenCycleMap: Docs*, 2013. http://www.opencyclemap.org/docs/. (Zitiert auf Seite 64)

[Ope13k] OpenStreetMap. *OpenCycleMap: OwnMap*, 2013. http://www.thunderforest.com/custom/. (Zitiert auf Seite 64)

[Ope13l] OpenStreetMap. *OpenCycleMap: Terms*, 2013. http://www.opencyclemap.org/terms/. (Zitiert auf Seite 65)

[Ope13m] Openstreetmap. *Osmarender: Frontend*, 2013. http://wiki.openstreetmap.org/wiki/Osmarender_Frontend. (Zitiert auf Seite 71)

[Ope13n] Openstreetmap. *Osmarender: Installation*, 2013. http://wiki.openstreetmap.org/wiki/Osmarender/Howto. (Zitiert auf Seite 72)

[Ope13o] Openstreetmap. *Osmarender: Symbole*, 2013. http://wiki.openstreetmap.org/wiki/Osmarender/Symbols. (Zitiert auf Seite 71)

[Ope13p] Openstreetmap. *Osmarender: Wartung, englische Version*, 2013. http://wiki.openstreetmap.org/wiki/Osmarender. (Zitiert auf Seite 71)

[Ope13q] Openstreetmap. *Pyrender: Weiterentwicklung*, 2013. http://wiki.openstreetmap.org/wiki/Pyrender. (Zitiert auf Seite 85)

[OSM13a] OSM2World. *OSM2World: Download*, 2013. http://osm2world.org/download/. (Zitiert auf Seite 83)

[OSM13b] OSM2World. *OSM2World: Main*, 2013. http://osm2world.org/. (Zitiert auf Seite 83)

[Osm13c] Osmand. *Osmand:Download*, 2013. http://code.google.com/p/osmand/downloads/list. (Zitiert auf Seite 70)

[Osm13d] Osmand. *Osmand:Wiki*, 2013. http://code.google.com/p/osmand/. (Zitiert auf Seite 70)

[SMR13a] SMRender. *SMRender: Blog*, 2013. https://www.cypherpunk.at/tag/smrender/. (Zitiert auf Seite 73)

[SMR13b] SMRender. *SMRender: Downloads*, 2013. http://www.abenteuerland.at/download/smrender/. (Zitiert auf Seite 74)

[SMR13c] SMRender. *SMRender: Rules*, 2013. http://www.abenteuerland.at/smrender/descr.html#tth_sEc5. (Zitiert auf Seite 73)

[Sol13a] Solocal Group. *Mappy: About*, 2013. http://corporate.mappy.com/. (Zitiert auf Seite 33)

[Sol13b] Solocal Group. *Mappy: Funktionen*, 2013. http://connect.mappy.com/en/home. (Zitiert auf Seite 33)

[Sol13c] Solocal Group. *Mappy: Karte*, 2013. http://de.mappy.com/. (Zitiert auf Seite 33)

[Tom13a] TomTom. *TomTom: Dokumentation*, 2013. http://developer.tomtom.com/. (Zitiert auf Seite 45)

[Tom13b] TomTom. *TomTom: Karte*, 2013. http://www.routes.tomtom.com/#. (Zitiert auf Seite 45)

[Tom13c] TomTom. *TomTom: Produkte*, 2013. http://www.tomtom.com/en_gb/licensing/products/maps/. (Zitiert auf Seite 45)

[Uni13] University of Heidelberg. *OSM3D: Karte*, 2013. http://www.osm-3d.org/map.htm. (Zitiert auf Seite 80)

Alle URLs wurden zuletzt am 17.07.2013 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift