CityJSON

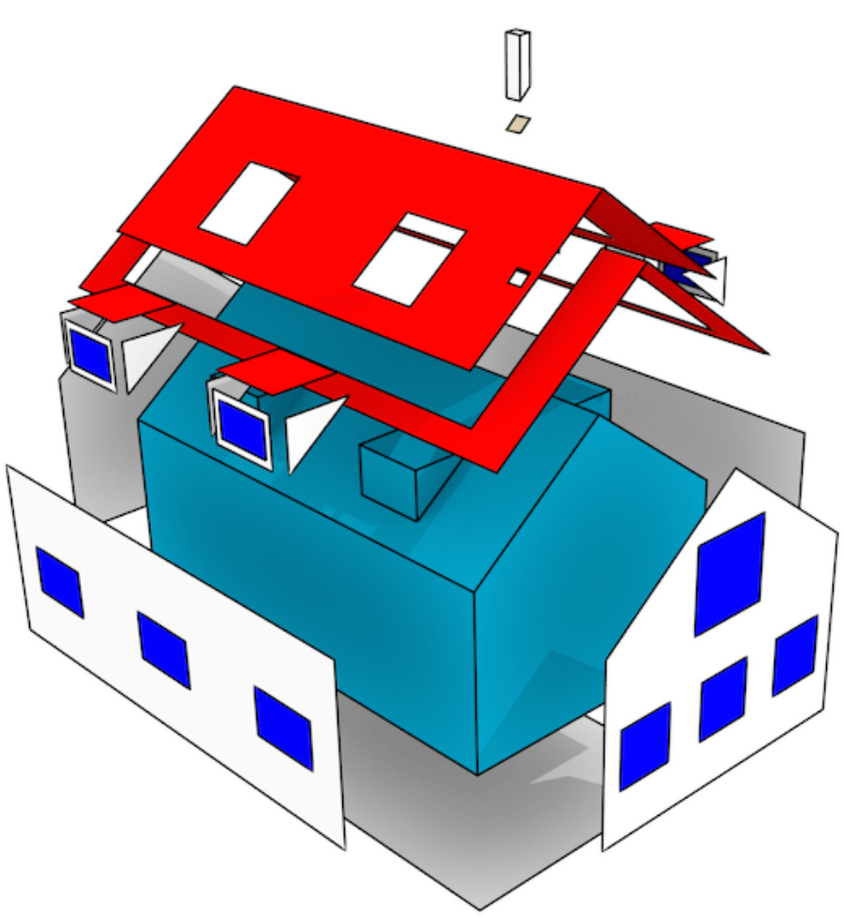
A compact and developer-friendly JSON-based encoding of the CityGML data model

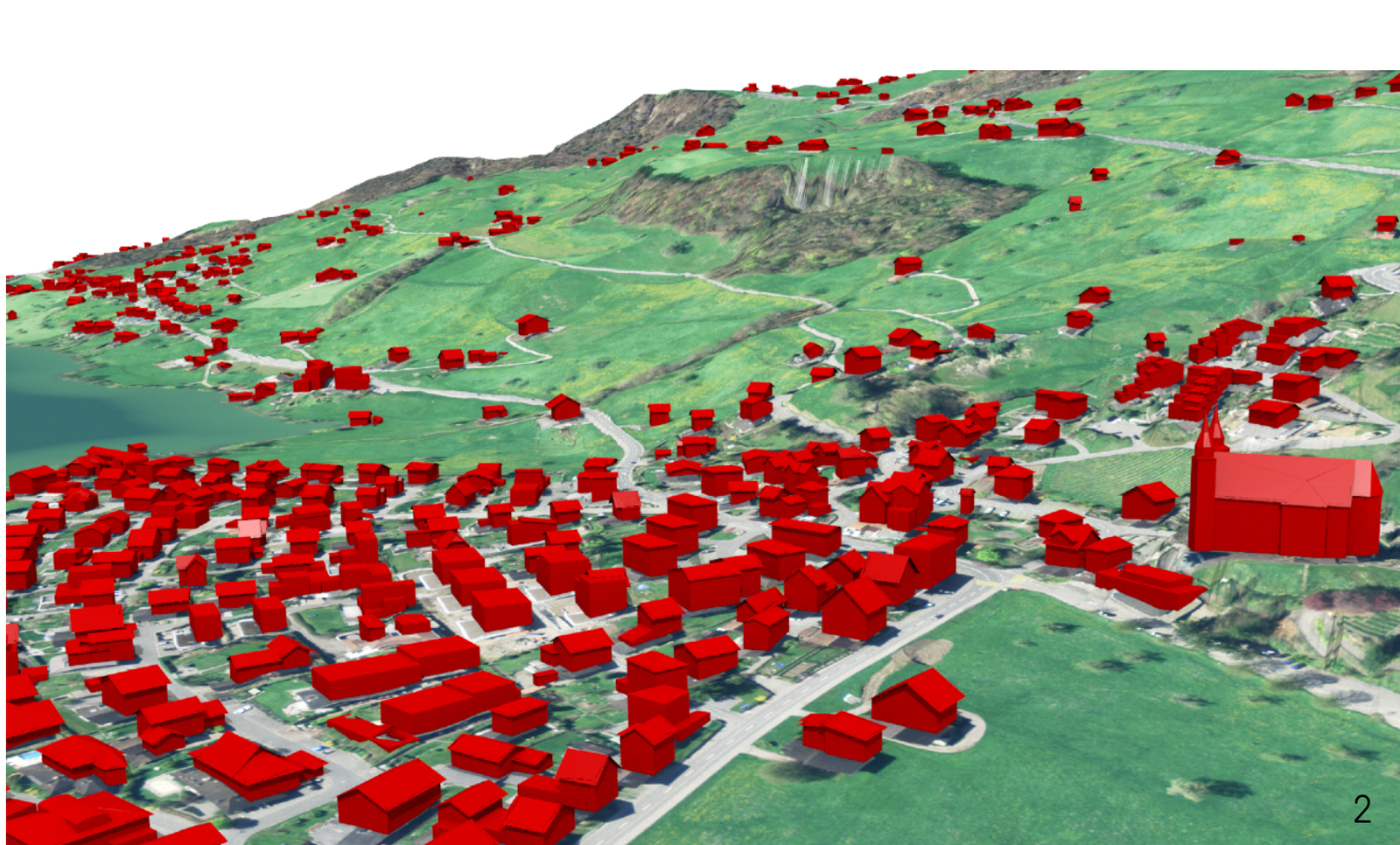
Hugo Ledoux

FOSS4GNL
Delft, the Netherlands
2019-06-20

CityGML

international standard (from OGC) for representing and storing 3D city models





CityGML files are very complex

- files are deeply nested, and large
- many “points of entry”
- many diff ways to do one thing

James Fee
@jamesmfee
Follow

I just got sent a CityGML file.

11:41 PM - Jun 29, 2016

1 2 8

➡ few software packages use CityGML

➡ no parsers in JavaScript

➡ I personally get 😩 each time I get a new file

CityGML

Data model

2 encodings

a GML encoding

CityJSON

Why an alternative encoding? Read this.

Ledoux *et al. Open Geospatial Data, Software and Standards* (2019) 4:4
https://doi.org/10.1186/s40965-019-0064-0

Open Geospatial Data, Software and Standards

ORIGINAL ARTICLE Open Access

CityJSON: a compact and easy-to-use encoding of the CityGML data model

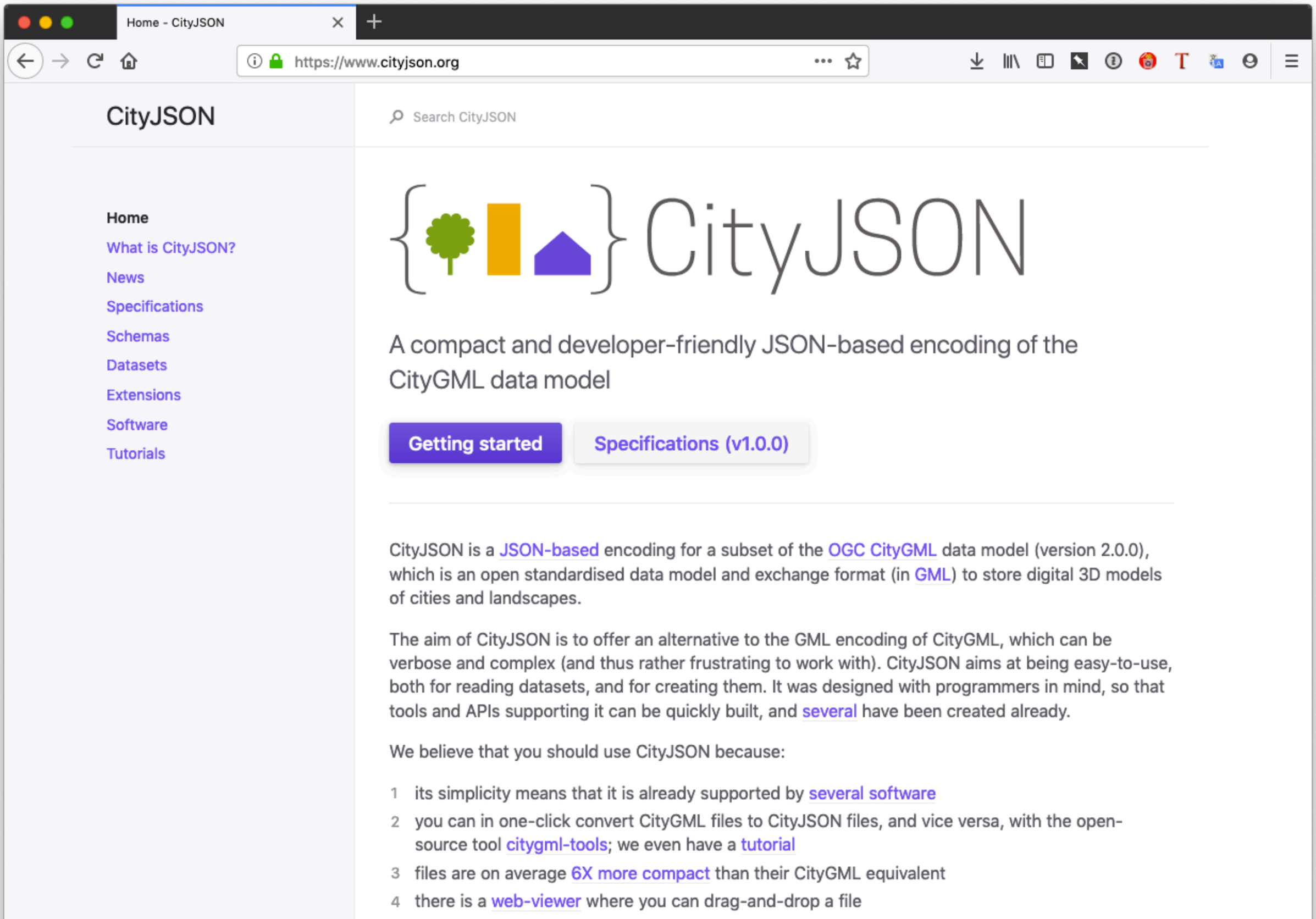
Hugo Ledoux*, Ken Arroyo Ohori, Kavisha Kumar, Balázs Dukai, Anna Labetski and Stelios Vitalis

Abstract

The international standard CityGML is both a data model and an exchange format to store digital 3D models of cities. While the data model is used by several cities, companies, and governments, in this paper we argue that its XML-based exchange format has several drawbacks. These drawbacks mean that it is difficult for developers to implement parsers for CityGML, and that practitioners have, as a consequence, to convert their data to other formats if they want to exchange them with others. We present CityJSON, a new JSON-based exchange format for the CityGML data model (version 2.0.0). CityJSON was designed with programmers in mind, so that software and APIs supporting it can be quickly built. It was also designed to be compact (a compression factor of around six with real-world datasets), and to be friendly for web and mobile development. We argue that it is considerably easier to use than the CityGML format, both for reading and for creating datasets. We discuss in this paper the main features of CityJSON, briefly present the different software packages to parse/view/edit/create files (including one to automatically convert between the JSON and GML encodings), analyse how real-world datasets compare to those of CityGML, and we also introduce *Extensions*, which allow us to extend the core data model in a documented manner.

Keywords: 3D city modelling, CityGML, Data modelling

CityJSON


Home - CityJSON
https://www.cityjson.org
CityJSON
Search CityJSON
Home
What is CityJSON?
News
Specifications
Schemas
Datasets
Extensions
Software
Tutorials
CityJSON
A compact and developer-friendly JSON-based encoding of the CityGML data model
Getting started
Specifications (v1.0.0)
CityJSON is a JSON-based encoding for a subset of the OGC CityGML data model (version 2.0.0), which is an open standardised data model and exchange format (in GML) to store digital 3D models of cities and landscapes.
The aim of CityJSON is to offer an alternative to the GML encoding of CityGML, which can be verbose and complex (and thus rather frustrating to work with). CityJSON aims at being easy-to-use, both for reading datasets, and for creating them. It was designed with programmers in mind, so that tools and APIs supporting it can be quickly built, and several have been created already.
We believe that you should use CityJSON because:
1 its simplicity means that it is already supported by several software
2 you can in one-click convert CityGML files to CityJSON files, and vice versa, with the open-source tool citygml-tools; we even have a tutorial
3 files are on average 6X more compact than their CityGML equivalent
4 there is a web-viewer where you can drag-and-drop a file
5 you can easily manipulate files with cjio, you can for instance merge files, remove/filter objects,

version 1.0.0 released recently

- all modules mapped 💪🏽
- software for full conversion CityGML <-> CityJSON
- several software already
- Extensions (ADEs) are possible

1. ”Building”
2. ”BuildingPart”
3. ”BuildingInstallation”
4. ”Road”
5. ”Railway”
6. ”TransportSquare”
7. ”TINRelief”
8. ”WaterBody”
9. ”PlantCover”
10. ”SolitaryVegetationObject”
11. ”LandUse”
12. ”CityFurniture”
13. ”GenericCityObject”
14. ”Bridge”
15. ”BridgePart”
16. ”BridgeInstallation”
17. ”BridgeConstructionElement”
18. ”Tunnel”
19. ”TunnelPart”
20. “TunnelInstallation”
21. “CityObjectGroup”

Same information as CityGML, but in JSON format

```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```

A CityJSON file

```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```

version CityJSON

A CityJSON file

```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```

metadata, ISO19115 "compliant"

CityGML has no mechanism in v2.0 but we thought it's important

ALL geometries have the same CRS, unlike CityGML

A CityJSON file


```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```


All City Objects listed here, *indexed* by their ID

Each have geometries + attributes

A CityJSON file

```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```

Geometry is ID of the vertex, global list

compression + more “topology”

A CityJSON file

```
{
  "type": "CityJSON",
  "version": "1.0",
  "metadata": {
    "referenceSystem": "urn:ogc:def:crs:EPSG::7415",
  },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": {
        "measuredHeight": 22.3,
        "roofType": "gable",
        "owner": "Elvis Presley"
      },
      "geometry": [
        {
          "type": "MultiSurface",
          "boundaries": [
            [[0, 3, 2, 1]], [[4, 5, 6, 7]], [[0, 1, 5, 4]]
          ]
        }
      ]
    }
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [111.1, 321.1, 12.0],
    ...
  ],
  "appearance": {
    "materials": [],
    "textures":[],
    "vertices-texture": []
  }
}
```

material + texture possible

BuildingParts: links between City Objects

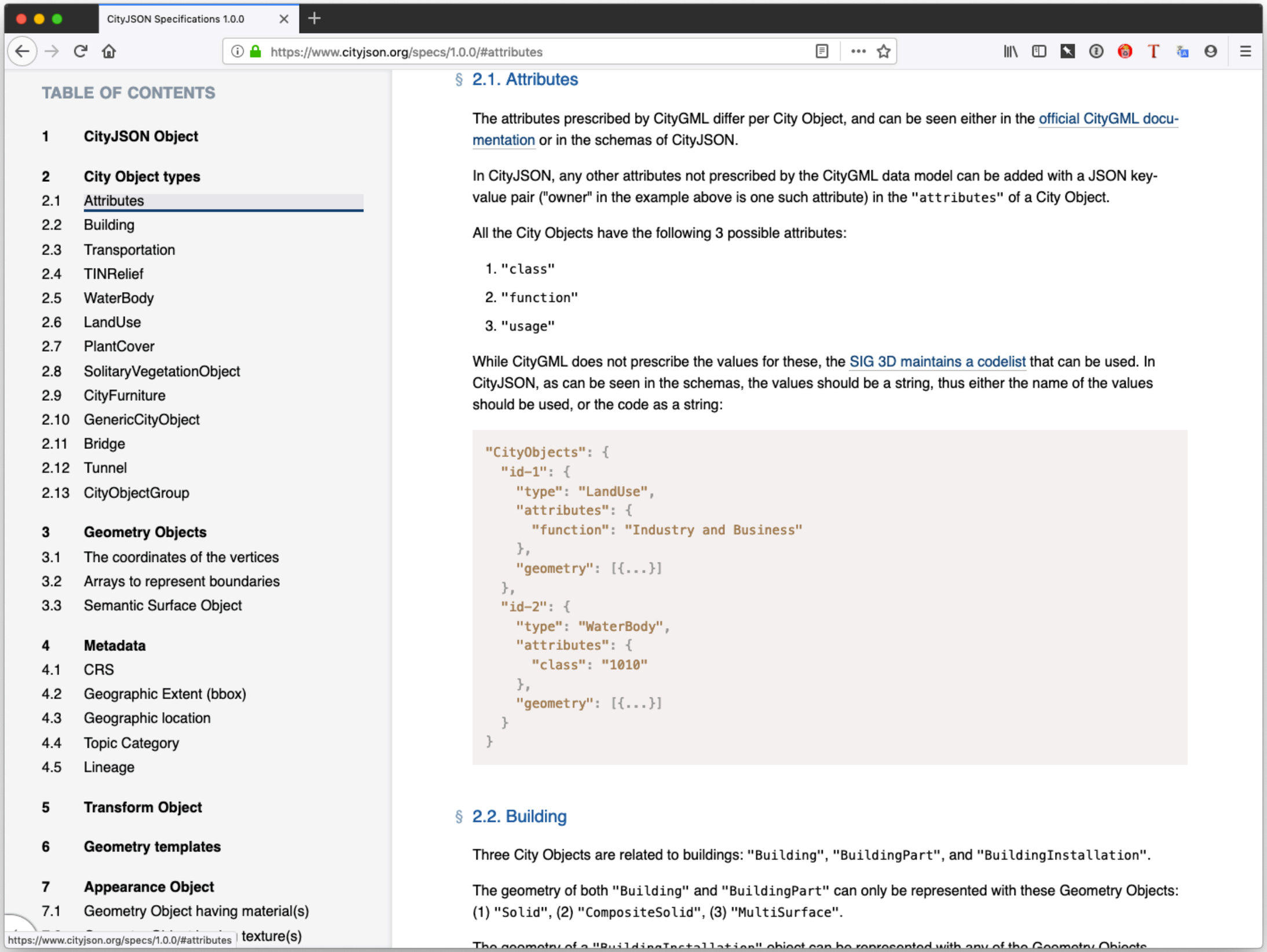
```
"CityObjects": {
  "id-1": {
    "type": "Building",
    "attributes": {
      "roofType": "gable"
    },
    "children": ["id-56", "id-832", "mybalcony"]
  },
  "id-56": {
    "type": "BuildingPart",
    "parents": ["id-1"],
    ...
  },
  "mybalcony": {
    "type": "BuildingInstallation",
    "parent": ["id-1"],
    ...
  }
}
```

BuildingParts: links between City Objects

```
"CityObjects": {
  "id-1": {
    "type": "Building",
    "attributes": {
      "roofType": "gable"
    },
    "children": ["id-56", "id-832", "mybalcony"]
  },
  "id-56": {
    "type": "BuildingPart",
    "parents": ["id-1"],
    ...
  },
  "mybalcony": {
    "type": "BuildingInstallation",
    "parents": ["id-1"],
    ...
  }
}
```

goal == a flat schema

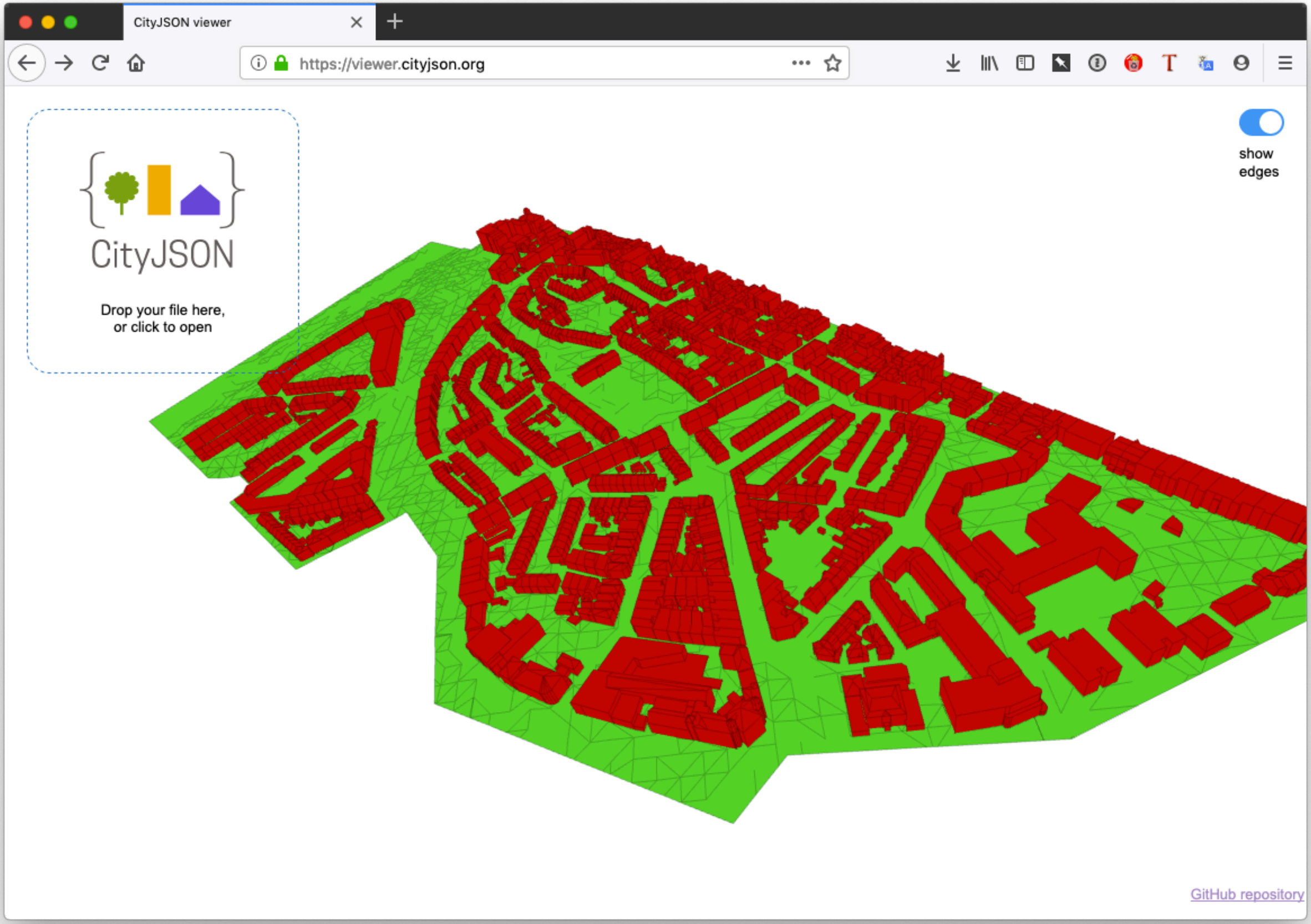
Specifications give all the gory details



CityJSON software?

We have that.

web-viewer


CityJSON viewer
https://viewer.cityjson.org
CityJSON
Drop your file here,
or click to open
show
edges
GitHub repository

azul

Featured Top Charts Categories Purchased Updates Search

Azul 4+

azul is a 3D viewer for 3D city models in (City)GML, CityJSON, OBJ, OFF and POLY. It supports loading multiple files, selecting objects by clicking them or selecting them in the sidebar, and browsing their attributes.

What's New in Version 0.8.1

Support for CityJSON 0.5

Install

Azul Support

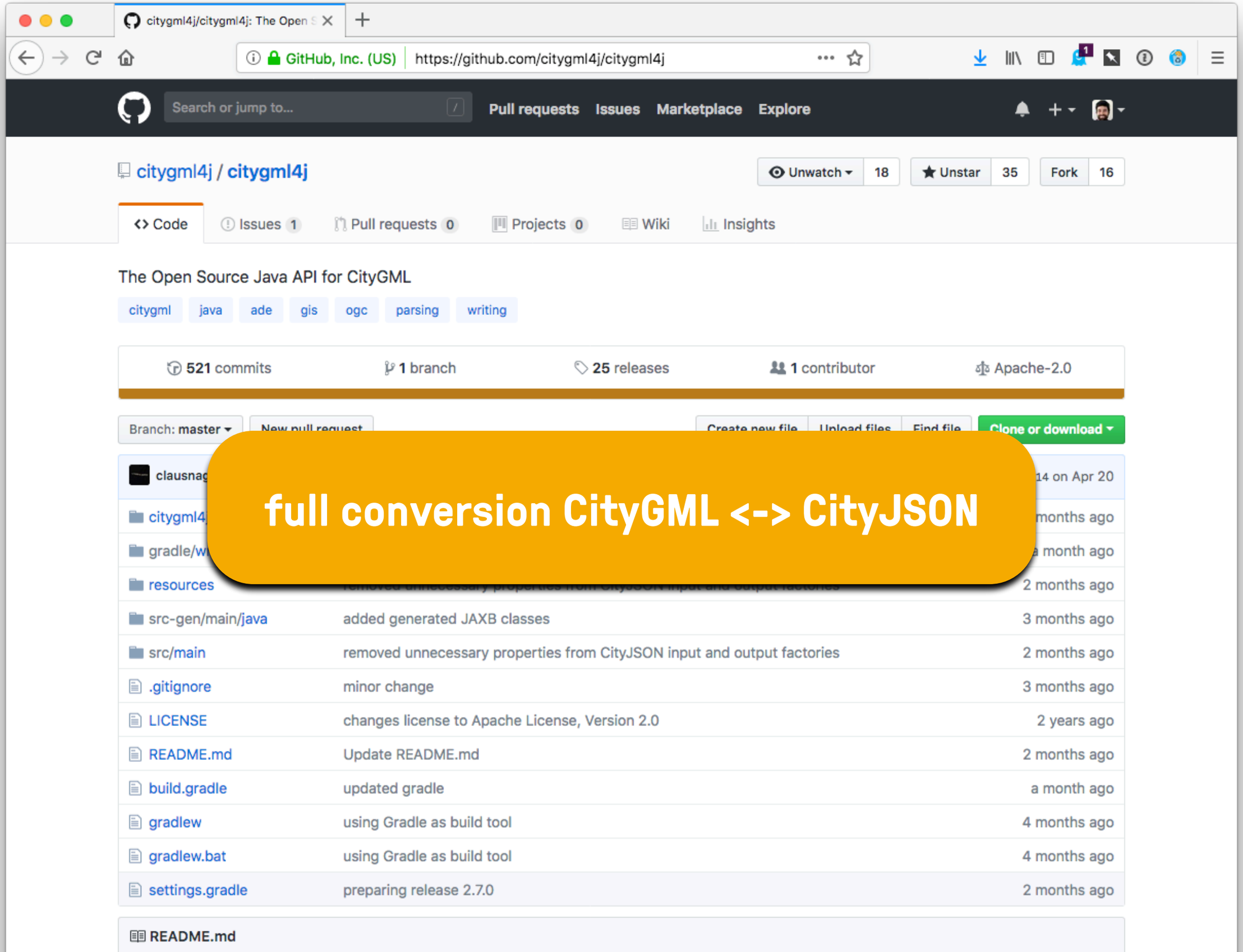
Information

Category: Utilities
Updated: 19 December 2017
Version: 0.8.1
Price: Free
Size: 9.6 MB
Family Sharing: Yes
Language: English
Developer: Ken Arroyo Ohori
© 2016-2017 Ken Arroyo Ohori

Rated 4+

Compatibility:
macOS 10.13 or later, 64-bit processor

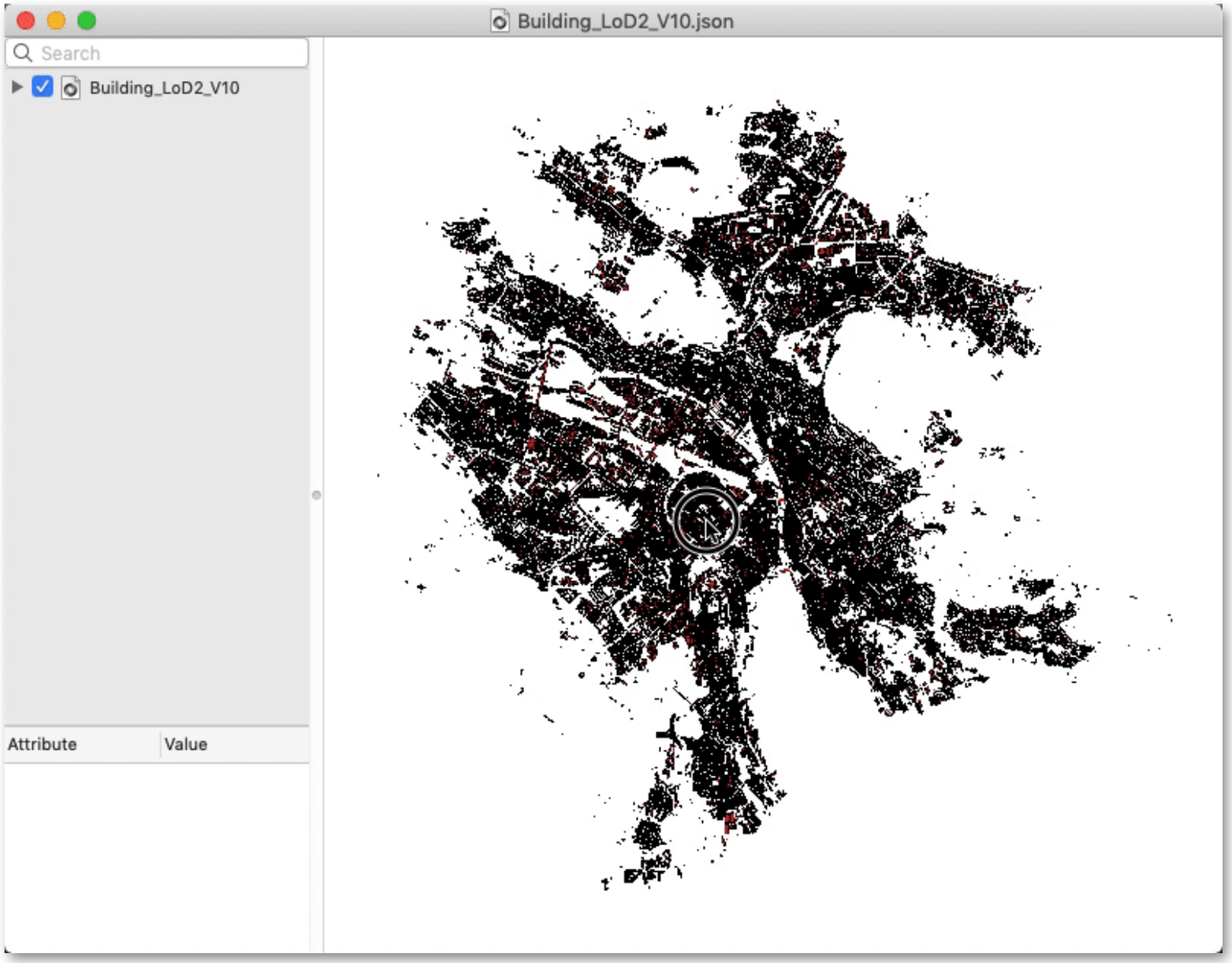
citygml4j


citygml4j/citygml4j: The Open S
GitHub, Inc. (US)
https://github.com/citygml4j/citygml4j
Search or jump to...
Pull requests
Issues
Marketplace
Explore
citygml4j / citygml4j
Unwatch 18
Unstar 35
Fork 16
Code
Issues 1
Pull requests 0
Projects 0
Wiki
Insights
The Open Source Java API for CityGML
citygml
java
ade
gis
ogc
parsing
writing
521 commits
1 branch
25 releases
1 contributor
Apache-2.0
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
clausnag
14 on Apr 20
citygml4j
months ago
gradle/w
month ago
resources
2 months ago
src-gen/main/java
added generated JAXB classes
3 months ago
src/main
removed unnecessary properties from CityJSON input and output factories
2 months ago
.gitignore
minor change
3 months ago
LICENSE
changes license to Apache License, Version 2.0
2 years ago
README.md
Update README.md
2 months ago
build.gradle
updated gradle
a month ago
gradlew
using Gradle as build tool
4 months ago
gradlew.bat
using Gradle as build tool
4 months ago
settings.gradle
preparing release 2.7.0
2 months ago
README.md
full conversion CityGML <-> CityJSON

Compression factor == ~6X

file	CityGML size (original)	CityGML size (w/o spaces)	textures?	CityJSON	CityJSON compressed	compression factor
CityGML demo "GeoRes"	4.3MB	4.1MB	yes	582KB	524KB	8.0
CityGML v2 demo "Railway"	45MB	34MB	yes	4.5MB	4.3MB	8.1
Den Haag "tile 01"	23MB	18MB	no, material	3.1MB	2.9MB	6.2
Montréal VM05	56MB	42MB	yes	5.7MB	5.4MB	7.8
New York LoD2 (DA13)	590MB	574MB	no	110MB	105MB	5.5
Rotterdam Delfshaven	16MB	15MB	yes	2.8MB	2.6MB	5.4
Vienna	37MB	36MB	no	5.6MB	5.3MB	6.8

One example: Zürich LoD2 buildings



CityGML = 3.0GB

(but 1GB of spaces/CRs/tabs!)

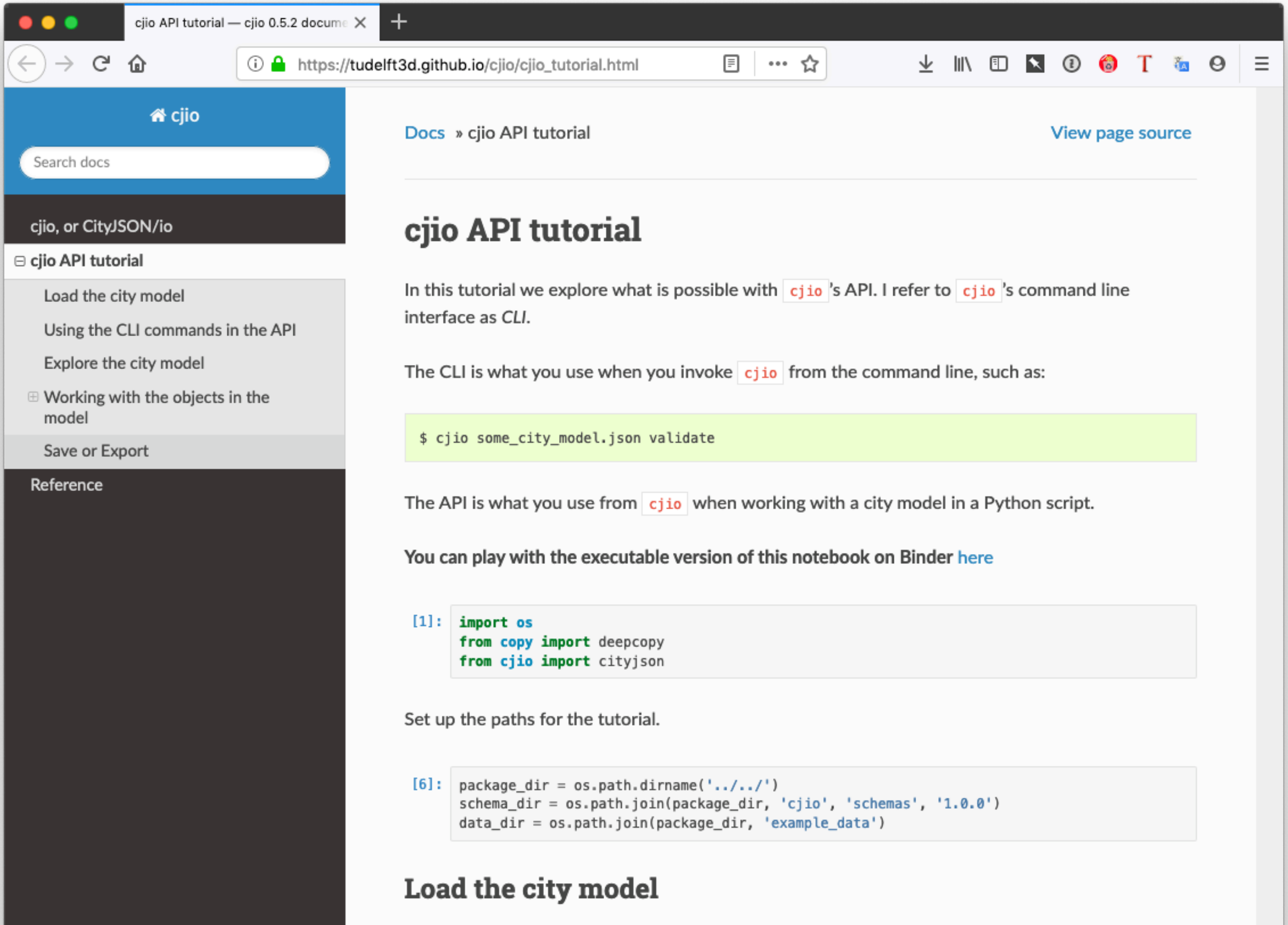
CityJSON = 292MB

Compression == 7.1X

Python parser is simple

```
import json

fin = open('mycity.json')
cm = json.loads(fin.read())

print "There are", len(cm['CityObjects']), "CityObjects"

# list all ids
for id in cm['CityObjects']:
    print "\t", id
```

Python API (work-in-progress)



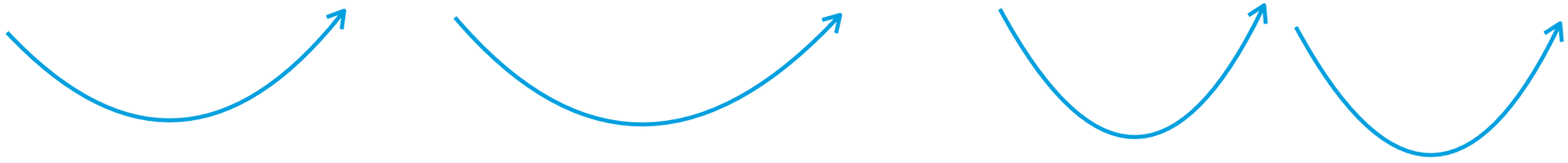
cjio (CityJSON/io)

2. bash

```
Hugos-MacBook-Pro:rotterdam hugo$ cjio
Usage: cjio [OPTIONS] INPUT COMMAND1 [ARGS]... [COMMAND2 [ARGS]...]...

  Process and manipulate a CityJSON file, and allow different outputs. The
  different operators can be chained to perform several processing in one
  step, the CityJSON model goes through the different operators.

  To get help on specific command, eg for 'validate':

      cjio validate --help

  Usage examples:

      cjio example.json validate
      cjio example.json remove_textures info
      cjio example.json subset --id house12 remove_materials save out.json

Options:
  --version                Show the version and exit.
  --off                    Load an OFF file and convert it to one CityJSON
                           GenericCityObject.
  --ignore_duplicate_keys  Load a CityJSON file even if some City Objects have
                           the same IDs (technically invalid file)
  --help                   Show this message and exit.

Commands:
  compress                   Compress a CityJSON file, ie stores its...
  decompress                 Decompress a CityJSON file, ie remove the...
  info                       Output info in simple JSON.
  merge                      Merge the current CityJSON with others.
  remove_duplicate_vertices  Remove duplicate vertices a CityJSON file.
  remove_materials           Remove all materials from a CityJSON file.
  remove_orphan_vertices     Remove orphan vertices a CityJSON file.
  remove_textures            Remove all textures from a CityJSON file.
  save                       Save the CityJSON to a file.
  subset                     Create a subset of a CityJSON file.
  update_bbox                Update the bbox of a CityJSON file.
  update_crs                 Update the CRS with a new value.
  validate                   Validate the CityJSON file: (1) against its...
```

cjio (CityJSON/io)

```
$ cjio myfile.json assign_epsg 7415 subset -cotype Buidling compress save out.json
```



pip install cjio

QuickTime Player File Edit View Window Help

Mon 17:35

denhaag

Name	Date Modified	Size
DenHaag_01_uncompressed.json	12/04/2018	3,3 MB
DenHaag_01.json	12/04/2018	3,1 MB
DenHaag_01.json.zip	12/04/2018	583 KB
DenHaag_01.nocr.xml	12/04/2018	18,6 MB
DenHaag_01.xml	12/04/2018	23,9 MB
DenHaag_01.xml.zip	12/04/2018	1 MB
v08	21/11/2018	--
val3dity.log	28/11/2018	833 KB

Favourites: Desktop, AirDrop, Applications, Documents, temp, hugo, Dropbox, recent, surfdrive

iCloud: iCloud Drive

Locations: hugo-mbp, Remote Disc, Hugo's Air..., Network

Tags: Gray, Orange, Green

1 of 8 selected, 18,78 GB available

Untitled Project - QGIS

Browser: Favorites, Home, /, /Volumes, GeoPackage

Layers

Value Tool

Enable

Table Graph Options

Decimals 2

Layer | Value

Recent Projects

p
/Users/hugo/teaching/geo1015_material/hw/03/mycode/p.qgz
USER:100055 (* Generated CRS (+proj=utm +zone=55 +south +ellps=GRS80 +units=m +no_de

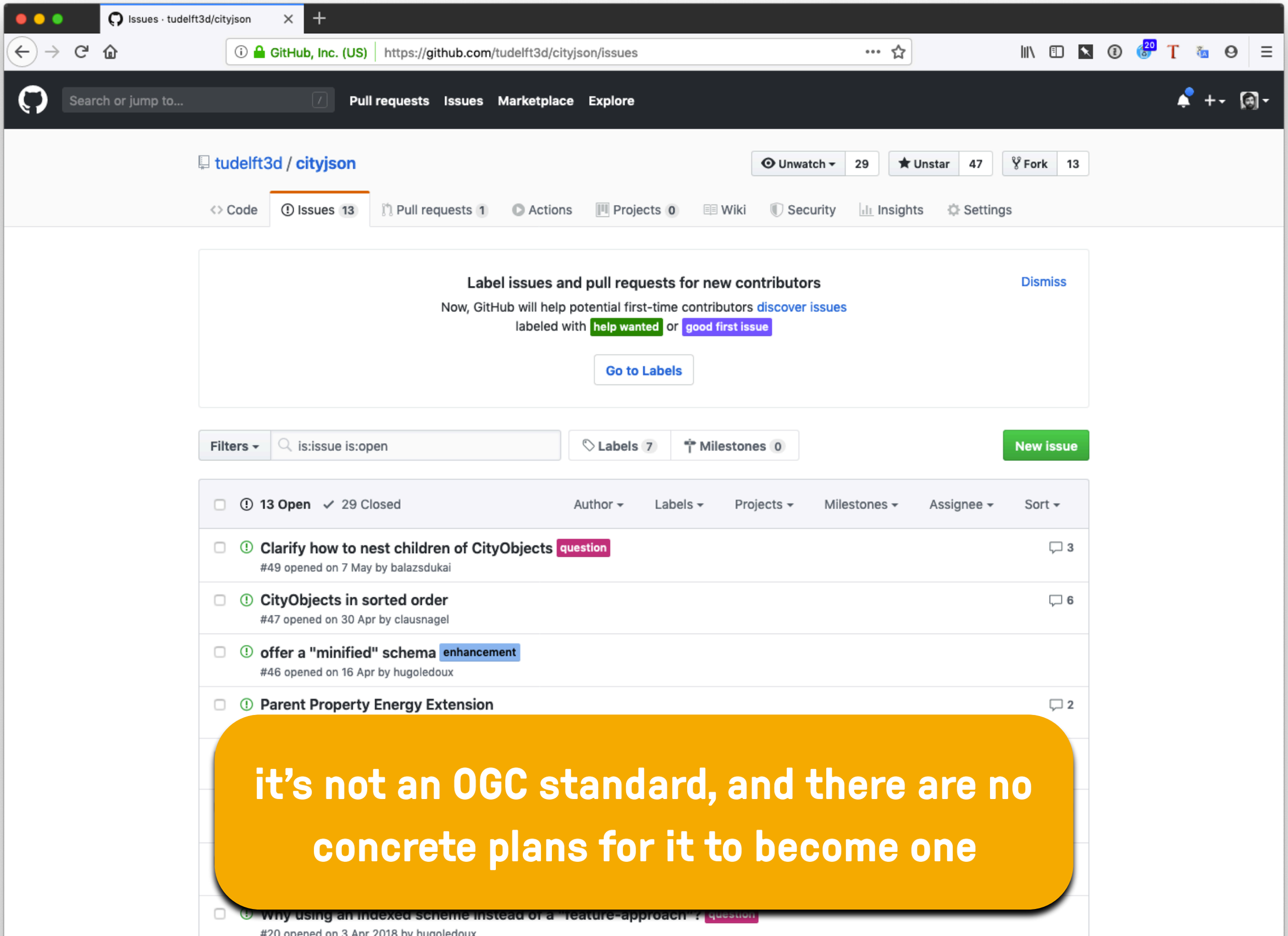
project
/Users/hugo/teaching/geo1015_material/hw/01/marking/project.qgz
USER:100045 (* Generated CRS (+proj=longlat +datum=WGS84 +no_defs))

viewshed
/Users/hugo/Dropbox/teaching/DTM/help-stuff/les12/viewshed.qgz
USER:100027 (* Generated CRS (+proj=tmerc +lat_0=0 +lon_0=15 +k=1 +x_0=3500000 +y_0=0

qgis_project
/Users/hugo/projects/3dfier/example_data/bgt/qgis_project.qgs

Type to locate (⌘K) Ready Coordinate Scale 1:22390718 Magnifier 100% Rotation 0,0 ° Render EPSG:4326

Help? Feedback? All development is open



Getting started?

Getting started with CityJSON - City

https://www.cityjson.org/tutorials/getting-started/

CityJSON

Search CityJSON

- Home
- What is CityJSON?
- News
- Specifications
- Schemas
- Datasets
- Extensions
- Software
- Tutorials
  - **Getting started with CityJSON**
  - Converting to/from CityGML files
  - Validation of a CityJSON file
  - Mapping the CityGML Noise ADE to a CityJSON Extension
  - Modelling guide for CityJSON

Tutorials / Getting started with CityJSON

Getting started with CityJSON

TABLE OF CONTENTS

Download a simple file with 2 buildings

Download twobuilding.json, a simple file with 2 buildings.

You can open that file in any text editor to see its structure, and notice that you can manually edit it to change values and/or add new buildings, new metadata, or delete some attributes.

twobuildings.json

```
{
    "type": "CityJSON",
    "version": "1.0",
    "metadata": {
        "geographicalExtent": [
            300578.235,
            5041258.061,
            13.688,
            300618.138,
            5041289.394,
            29.45
        ]
    },
    "CityObjects": {
        "Building 1": {
```

thank you.

Hugo Ledoux
h.ledoux@tudelft.nl
3d.bk.tudelft.nl/hledoux

CityJSON
https://cityjson.org

TU Delft Delft University of Technology