Updates on the CityJSON Energy Extension

Hugo Ledoux & Kavisha

2018-12-07
10th Workshop on the CityGML Energy ADE
Delft, the Netherlands

TU Delft
3Dgeoinfo

CityGML

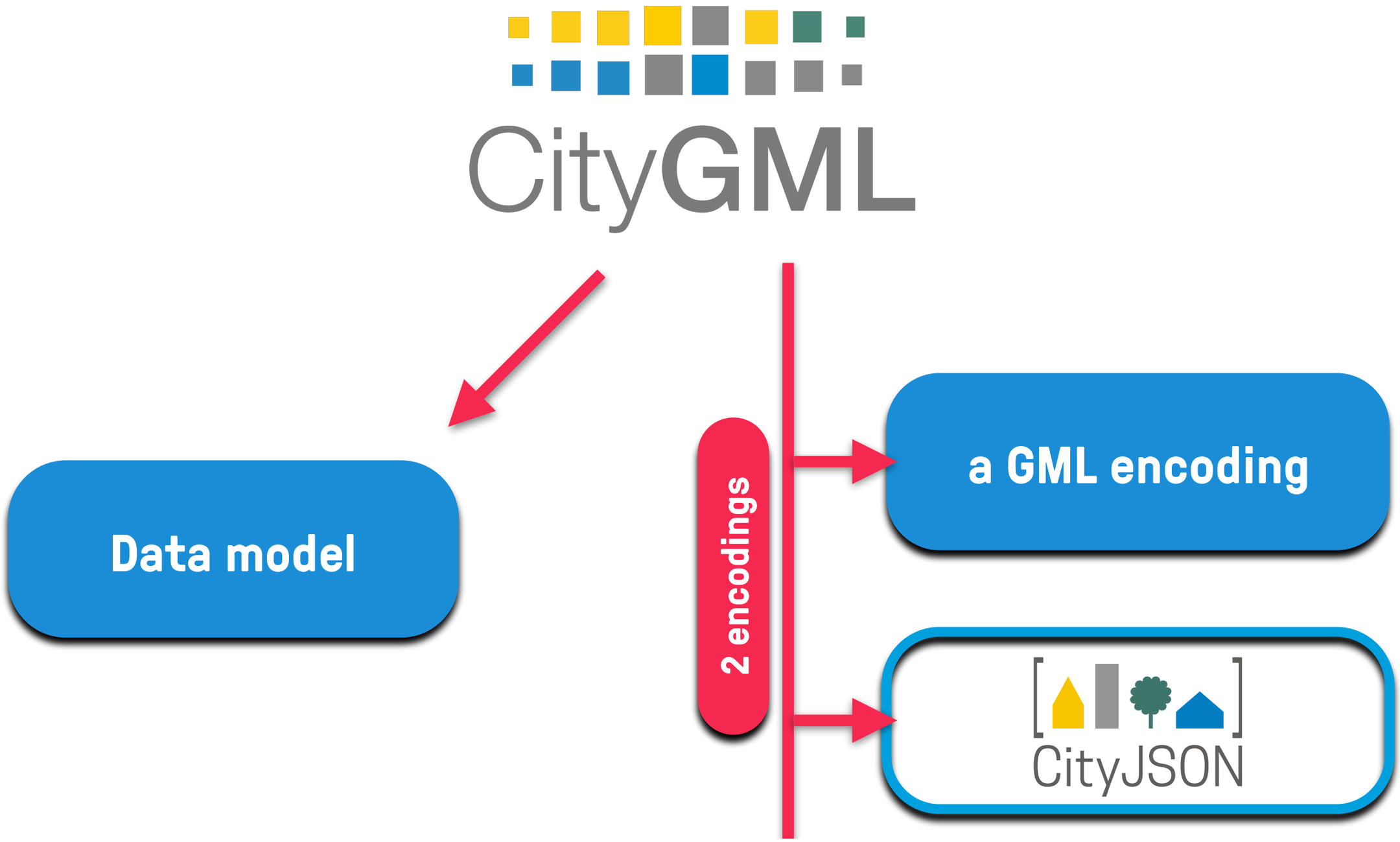
Data model

a GML encoding

both are called CityGML 🤷🏼‍♂️

CityGML files are very complex

- CityGML files are complex and large
- No parsers in JavaScript
- Few software packages use CityGML
- Many diff ways to do one thing
- I personally get 😩 each time I get a new file


CityGML
Data model
2 encodings
a GML encoding
CityJSON

CityJSON

CityJSON — CityJSON 0.8 docu

https://www.cityjson.org/en/0.8/

CityJSON

CityGML without the GML

Watch 26

Navigation

CityJSON specifications
Schema validation
Software
Extensions
CityGML support
Example datasets
Changelog
Contact

CityGML homepage
JSON homepage
cjio (CityJSON/io)
GitHub repository

Quick search

Go

CityJSON

(version 0.8)

```
{
  "type": "CityJSON",
  "version": "0.8",
  "extensions": {
    "+NoiseBuilding": "https://someurl.org/noise.json",
  },
  "metadata": { "referenceSystem": "urn:ogc:def:crs:EPSG::7415" },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": { "roofType": "gable" },
      "geometry": [{
        "type": "Solid",
        "lod": 2,
        "boundaries": []
      }]
    },
    "id-56": {...}
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [14.0, 2299.5, 14.0],
    ...
  ],
  "appearance": {
    "textures": []
  },
  "geometry-templates": {}
}
```

v0.8 released

- all modules mapped 💪🏽
- software for full conversion CityGML <-> CityJSON
- Extensions (ADEs in CityGML world) are possible
- software to visualise (azul) and to manipulate (cjio)

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”: “0.8”,
  ”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": "0.8",
  "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": "0.8",
  "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": "0.8",
  "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": "0.8",
  "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": "0.8",
  "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”,
    "parent": “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",
    "parent": "id-1",
    ...
  },
  "mybalcony": {
    "type": "BuildingInstallation",
    "parent": "id-1",
    ...
  }
}
```

goal == a flat schema

software

Python parser is simple

```python
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
```

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

full conversion CityGML <-> CityJSON

citygml4j		months ago
gradle/w		a month ago
resources	removed unnecessary properties from CityJSON input and output factories	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

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: mast r download

clausnag 14 on Apr 20

Watch out: only for the core modules not for ADEs

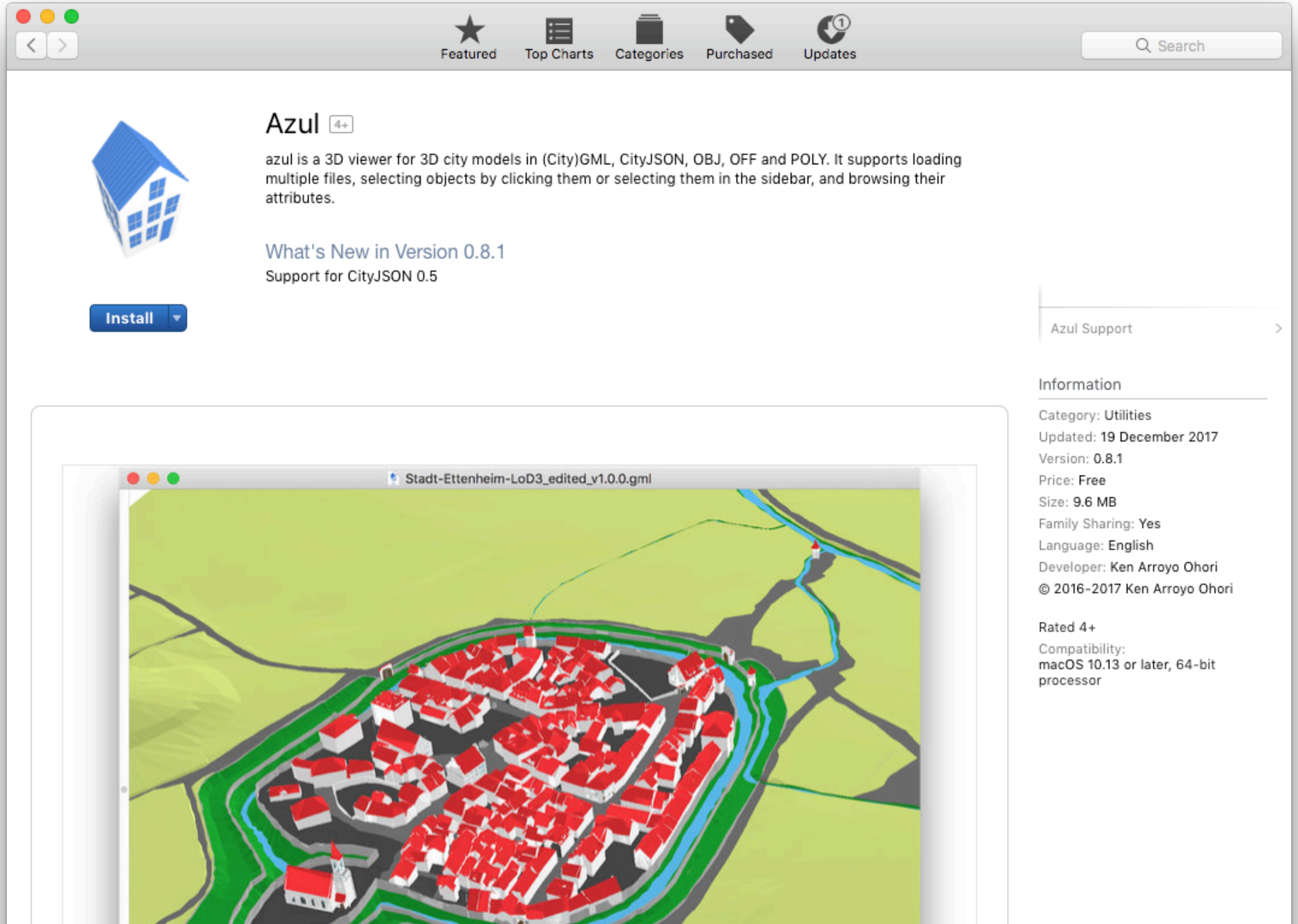
citygml4j		months ago
gradle/w		month ago
resource		months ago
src-gen/m		3 months ago
src/main	removed unnecessary properties from CityJSON input and output factories	2 months ago
.gitignore	minor change	3 months ago
LICENSE	changes license to Apache License, Version 2.0	2 years ago
README.md	Update 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

Compression factor == ~7X

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

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
Stadt-Ettenheim-LoD3_edited_v1.0.0.gml

cjio (CityJSON/io)

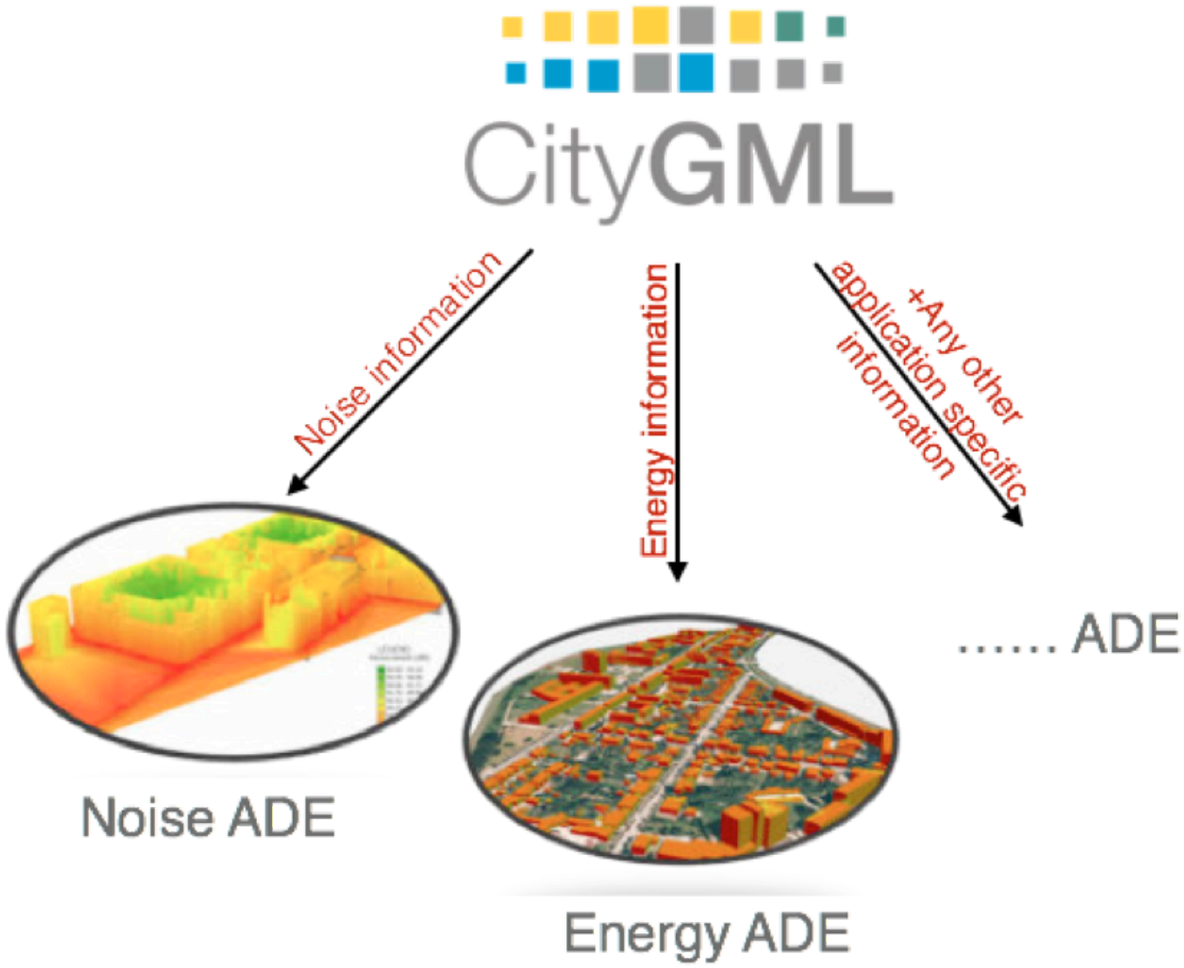
2. bash

```
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...
```

Extensions

(aka as ADEs)

CityGML **A**pplication **D**omain **E**xtensions



- https://www.citygml.org/ade/
- many designed in UML and with schemas
- but little software support...

CityJSON Extensions

- in beta in v0.9
- simpler than ADEs: strict rules for extensions
- less flexible than ADEs though
- one can add:
  1. new root properties (not much use)
  2. new attributes for existing City Objects
  3. new CityObjects
- one just defines these in a structured JSON file
- all downstream software should work (touch wood...)

Everything is documented

Extensions — CityJSON 0.9 bet

https://www.cityjson.org/en/development/exten

CityJSON

CityGML without the GML

Watch 28

Navigation

CityJSON specifications
Schema validation
Software
Extensions
- CityJSON Extensions
- Mapping of the Noise ADE to a CityJSON Extension
- Validation of files containing extensions

CityGML support
Example datasets
Changelog
Contact

CityGML homepage

Extensions

(version 0.9 beta)

- CityJSON Extensions
  - The Extension file
  - 1. Adding new complex attributes to existing City Objects
  - 2. Creating/extending new City Objects
  - 3. Adding new properties at the root of a document
- Mapping of the Noise ADE to a CityJSON Extension
  - Creating new City Objects
  - Adding complex types for CityFurniture
- Validation of files containing extensions

The CityGML data model allows us to represent the most common city features, but sometimes practitioners may want to model additional features and/or add certain attributes to the data model. For this, CityGML has the concept of ADEs (Application Domain Extensions). An ADE is defined in an extra XML Schema (XSD file) with its own namespace, and often inheritance is used to refine the classes of the CityGML data model, to define entirely new classes, and to modify any class by adding for instance new geometries and complex attribute anywhere in a City Model. The ADE allows us to document in a structured way, and also to validate, an instance of a CityGML document that would contain both classes from the core model and from the ADEs.

CityJSON Extensions

CityJSON

CityGML without the GML

Watch 28

Navigation

CityJSON specifications
Schema validation
Software
Extensions
CityGML support
Example datasets
Changelog
Contact

CityGML homepage
JSON homepage
cjio (CityJSON/io)
GitHub repository

Quick search

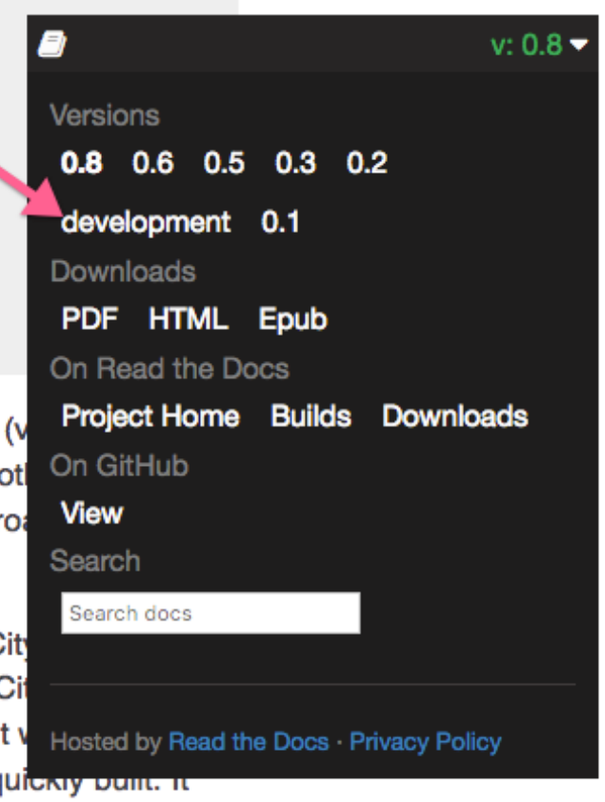
Go

CityJSON

(version 0.8)

```
{
  "type": "CityJSON",
  "version": "0.8",
  "extensions": {
    "+NoiseBuilding": "https://someurl.org/noise.json",
  },
  "metadata": { "referenceSystem": "urn:ogc:def:crs:EPSG::7415" },
  "CityObjects": {
    "id-1": {
      "type": "Building",
      "attributes": { "roofType": "gable" },
      "geometry": [{
        "type": "Solid",
        "lod": 2,
        "boundaries": []
      }]
    },
    "id-56": {...}
  },
  "vertices": [
    [23.1, 2321.2, 11.0],
    [14.0, 2299.5, 14.0],
    ...
  ],
  "appearance": {
    "textures": []
  },
  "geometry-templates": {}
}
```

to see the Extensions latest status online



CityJSON is a format for encoding a subset of the CityGML data model (v
using JavaScript Object Notation (JSON). A CityJSON file represents bot
try and the semantics of the city features of a given area, eg buildings, ro
vegetation, and the city furniture.

The aim of CityJSON is to offer an alternative to the GML encoding of Cit
can be verbose and complex (and thus rather frustrating to work with). Ci
at being easy-to-use, both for reading datasets, and for creating them. It
with programmers in mind, so that tools and APIs supporting it can be quickly built. It

www.cityjson.org

(all dev/discussions are open on GitHub)