



GeoDDS

een “geospatial rule engine”

Marco Duiker



Marco Duiker – *MD-kwadraat*

Sinds 1996 actief in het geo-werkveld

Sinds 2007 zelfstandig professional

Sinds 2009 deelnemer OpenGeoGroep

Alles geo!



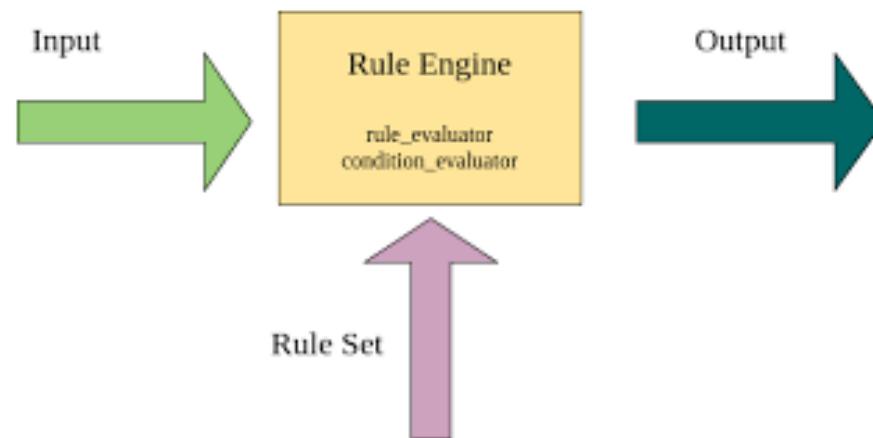
GEO ACADEMIE

**OPEN GEO
GROEP**
OG
G

 *MD-kwadraat*
Alles geo!

geoDSS

- ~~Ground Based Electro Optical Deep Space Surveillance~~
- Geospatial rule engine
 - Automatisch toetsen van een onderwerp (locatie met eigenschappen) tegen een set regels met als resultaat een rapportage



geoDSS

Wil ik hier wonen?

Met dit formulier kunt u zien of de plek die u invoert aantrekkelijk is om te wonen.

Subject

Huisnummer

Postcode

Geinteresseerd in:

Mannen

Vrouwen

Beide

Kinderhater



MD-kwadraat
Alles geo!

geoDSS

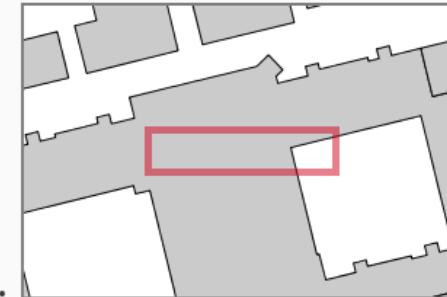
Results

Adreszoeker

Zoekt adres in de Basisregistratie Adressen en Gebouwen en zet het adres op de kaart

Found Zadelmakerstraat, 1315AK Almere on location: '[SRID=28992;POINT\(143355.815 486772.149\)](#)'

Kaart uit de Basisregistratie Adressen en Gebouwen



Gemeente

- Adres is gelegen in de gemeente Almere

Buurt

- Adres is gelegen in de buurt: Centrum Almere Stad

Weinig kinderen

Teveel kinderen (>15% voor een kinderhater)?

Uitkomst test: negatief



MD-kwadraat
Alles geo!

Huh, hoe?

Invoerscherm (subject):

```
title: Wil ik hier wonen?  
url: "http://localhost:8000/cgi-bin/interfaces.py"  
rule_set_file: "../geoDSS/examples/rule_sets/wonen.yaml"  
description: "Met dit formulier kunt u zien of de plek die u invoert aantrekkelijk is om te wonen."  
subtitle: "Subject"  
form_fields:  
- huisnummer:  
  name: huisnummer  
  label: Huisnummer  
  type: text  
- postcode:  
  name: postcode  
  label: Postcode  
  type: text  
- geslacht:  
  name: geslacht  
  label: "Geïnteresseerd in:"  
  type: radio  
  values:  
    - Mannen: man  
    - Vrouwen: vrouw  
    - Beide: beide  
- kinderhater:  
  name: kinderhater  
  label: Kinderhater  
  type: checkbox
```



MD-kwadraat
Alles geo!

Huh, hoe?

Rule set (regels)

```
- test:  
  type: processors.pdok_locatieserver  
  title: Adres zoeker  
  description: Zoekt adres in de Basisregistratie Adressen en Gebouwen en zet het adres op de kaart  
  url: "https://geodata.nationaalgeoregister.nl/locatieserver/v3/free?q="  
  report_template: "Found _{address}_ on location: ['subject.geometry'](https://bagviewer.kadaster.nl/lvbag/bag-viewer/index.html#?geometry.x={x}&ge  
  break_on_error: True  
  
- BAG map:  
  type: tests.get_map  
  title: Kaart uit de Basisregistratie Adressen en Gebouwen  
  description: ""  
  url: "https://geodata.nationaalgeoregister.nl/bag/wms/"  
  params:  
    layers: pand  
    styles: ""  
    width: 300  
    height: 200  
    format: "image/png"  
    version: "1.1.0"  
  buffer: 75  
  report_template: "![Centered_map_small]({url})"
```



MD-kwadraat
Alles geo!

Huh, hoe?

Rule set (regels)

```
- gemeente_naam:  
  type: tests.wfs2_SpatialOperator  
  title: Gemeente  
  description: ""  
  url: https://geodata.nationaalgeoregister.nl/bestuurlijkgrenzen/wfs?  
  typenames:  
    - "bestuurlijkgrenzen:gemeenten"  
  geometryname: geom  
  namespace:  
    prefix: app  
    URI: "http://www.deegree.org/app"  
  spatial_operator: DWithin  
  distance: 1  
  report_template: Adres is gelegen in de gemeente *{gemeentenaam}*  
  
- buurt:  
  type: tests.postgis_spatial_select  
  title: Buurt  
  description: ""  
  db:  
    dbname: gisdefault  
  schema: public  
  table: buurt  
  relationship: ST_Within  
  parameters:  
    - "subject.geometry"  
    - "wkb_geometry"  
  report_template: "Adres is gelegen in de buurt: *{bu_naam}*"
```



MD-kwadraat
Alles geo!

Voor wie?

- In te richten en te beheren door een GIS-medewerker
- Te gebruiken door een niet-GISser
- Uitbreidbaar door een knutselaar (of programmeur)



*MD-kwadraat
Alles geo!*



- ② Integreerbaar via:

- ② webservice (cgi, wsgi) (server meegeleverd)
- ② Commandoregel
- ② Scheduler
- ② Batch
- ② Python: import geoDSS

- ② Veel voorbeelden



*MD-kwadraat
Alles geo!*

Wat zit er allemaal in?

- Ui-generators

- Webformulier

- Loaders (voor rulesets)

- yaml
 - Json
 - Python
 - csv

- Processors

- Geocoders
 - Postgis bewerkingen (buffer)
 - Random point
 - Random value



MD-kwadraat
Alles geo!

Wat zit er allemaal in?

② Tests

- ② WFS2 spatial select
- ② WMS GetMap
- ② Postgis
- ② Requests
- ② Key-value compare
- ② Evaluate: Combineren van tests (and or xor ...)
- ② Remark

② Reporters

- ② Md: Markdown, html
- ② CSV



MD-kwadraat
Alles geo!

Nog meer?

Documentation

The [API-docs](#) provide useful information for both users and developers.

Especially useful are:

- | <https://marcoduiker.github.io/geoDSS/geoDSS/docs/API/interfaces.m.html>
- | <https://marcoduiker.github.io/geoDSS/geoDSS/docs/API/base.m.html>
- | <https://marcoduiker.github.io/geoDSS/geoDSS/docs/API/tests/index.html>
- | <https://marcoduiker.github.io/geoDSS/geoDSS/docs/API/processors/index.html>

Help you can find in the [help pages](#).



*MD-kwadraat
Alles geo!*

Nog meer?

Welcome to geoDSS's documentation!

Contents:

- [Introduction](#)
 - [So what is geoDSS and who is it meant for?](#)
 - [Getting help](#)
 - [Concepts](#)
 - [Getting started](#)
- [Install](#)
 - [Before you start](#)
 - [Basic](#)
 - [Advanced](#)
- [Defining a rule set](#)
 - [Heading](#)
 - [Rules](#)
- [Defining a subject](#)
- [Use Cases](#)
 - [Address in area's of interest](#)
 - [Batch geocode addresses](#)
 - [Monitor Webservices](#)
- [Loaders](#)
 - [yaml_loader](#)
 - [json_loader](#)
 - [python_loader](#)
- [Tests](#)
 - [unit_test](#)
 - [remark](#)
 - [get_map](#)
 - [key_value_compare](#)
 - [evaluate](#)
 - [request](#)

geoDSS.tests.remark module

[SHOW SOURCE](#)

Classes

class remark

This test always evaluates to True, regardless of the subject. This is useful for adding comments or remarks to the report.

The keys of the subject can be reported via parameter substitution as well as a timestamp.

Definition

definition is expected to be a dict having at least:

report_template (string): String to be reported when the test is True (which is always the case). The following placeholders will be replaced:

- {parameter} where parameter is a key in the subject will be replaced by the string representation of the value.
- {timestamp} will be replaced by the ISO timestamp.

Rule example

a useful yaml snippet for this test would be:

```
rules:  
    unit_test:  
        type: tests.remark  
        title: Remember  
        description: Remember  
        report_template: Remember the milk
```

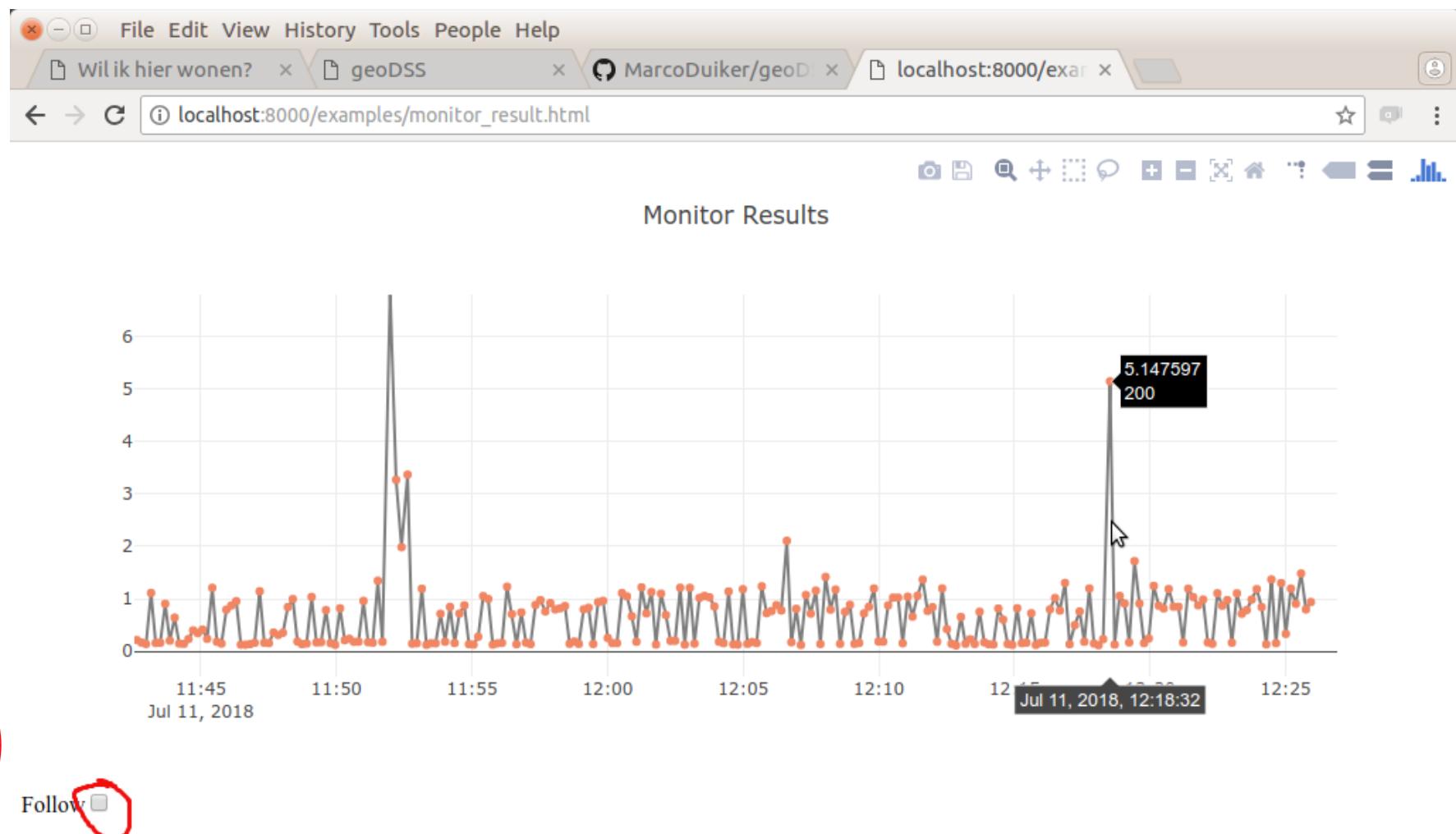
Subject Example

a useful subject for this would be (and any other subject will do as well):

```
subject = {}
```

Wat kun je dan?

- Automatisch toetsen
- Batch geocoderen
- Monitoring



Hoe verder?

- ② Meer degelijkheid ...
- ② Meer functionaliteit
 - ② OGR tests en processors
 - ② WFS 1.1 test (flexibel queries bouwen)
 - ② Meer manieren om tests te combineren
 - ② Alle ruwe testresultaten mee in het subject
 - ② PDF-uitvoer
 - ② Adres-op-kaart prikker
- ② Mooi(e) praktijkvoorbeeld(en)



*MD-kwadraat
Alles geo!*

Gebruiken?

- ❷ <https://github.com/MarcoDuiker/geoDSS>

The screenshot shows the GitHub repository page for 'MarcoDuiker / geoDSS'. The repository has 55 commits, 2 branches, 3 releases, and 2 contributors. The latest commit was 20 days ago. The README.md file contains a brief description of geoDSS as a simple yet extendable library for automatic testing of geospatial rules.

No description, website, or topics provided.

MarcoDuiker / geoDSS

55 commits 2 branches 3 releases 2 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

MarcoDuiker Improved on docs some more

geoDSS Improved on docs some more

.gitattributes updated docs and added docs generator

.gitignore first commit

.nojekyll first commit

LICENSE first commit

MANIFEST.in first commit

README.md Improved on docs some more

requirements.txt Improved on docs some more

setup.cfg first commit

setup.py add scheduling reporter, wfs2 test, bug fixes, help

README.md

geoDSS

geoDSS is a simple yet very extendable library to create automatic testing of a subject (argument) against rules and then report on the results.

geoDSS is a simple rule engine with a focus on geospatial rules.



MD-kwadraat
Alles geo!

en Bijdragen?

- ② <https://github.com/MarcoDuiker/geoDSS>
- ② Alles is makkelijk uitbreidbaar
 - ② Voor programmeurs
 - ② Voor knutselaars (met knippen en plakken)

```
def execute(self, subject):  
    '''  
    Executes the test.  
  
    'subject' is expected to be a dict.  
    (contents doesn't matter)  
    ...  
  
    result = self.definition["report_template"]  
    for key, value in subject.items():  
        result.replace('{'+key+'}', str(value))  
    result.replace('{timestamp}', datetime.datetime.now().isoformat())  
  
    self.logger.debug('Adding to the report: %s' % result)  
  
    self.decision = True  
  
    self.result.append(result)  
    self.executed = True  
  
    return self._finish_execution(subject)  
  
    # we have the report_template available to manipulate  
    # in this case we replace place holders by values in the subject  
  
    # you have a logger available to log some messages  
    # the loggers properties are set up in the rule_set definition  
  
    # don't forget to set self.decision to True,  
    # otherwise "Test decision is: False" is added to the report instead of the following:  
    # in this way we add a string to the report  
  
    # don't forget to set self.executed to True,  
    # otherwise "Error: test is not executed:" will be added to the report as well  
    # Returns False to end execution or the subject to continue to the next test
```



Conclusie

- ② Automatiseren van toetsingsprocessen
 - ② Door niet-techneuten
 - ② Flexibel
 - ② Goed gedocumenteerd
- ② Voor techneuten
 - ② Makkelijk te integreren in een omgeving
 - ② Makkelijk uit te breiden
 - ② Goed gedocumenteerd



*MD-kwadraat
Alles geo!*