FOSS4G-fr 2016

QGIS Server
Plugins et API Python

3Liz SARL

Création en Mars 2007

QGIS / LizMap / QGIS Server

Cadastre / QuickOSM / LayerBoard

FOSS4G-fr 2016

QGIS Server
Plugins et API Python

QGIS Server

Origines de QGIS Server

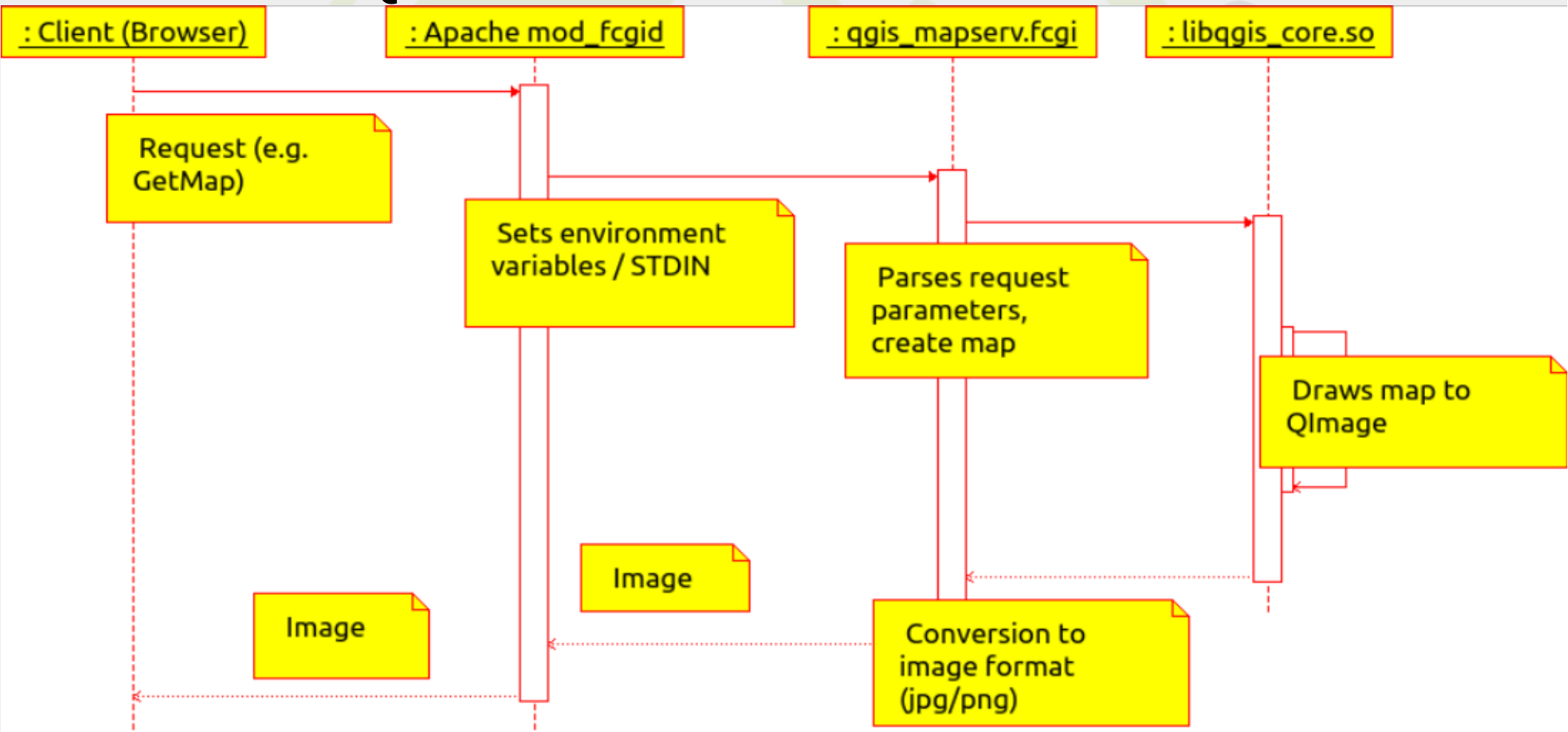
- Lancé en 2006 :
  - Au sein de projets de recherche
    - 'Orchestra' (Infrastructure de données spatiales européennes pour la gestion des risques)
    - 'SANY' (Sensor Anywhere)
  - Institute of Cartography (ETH Zurich)
  - Marco Hugentobler (SourcePole)

Origines de QGIS Server

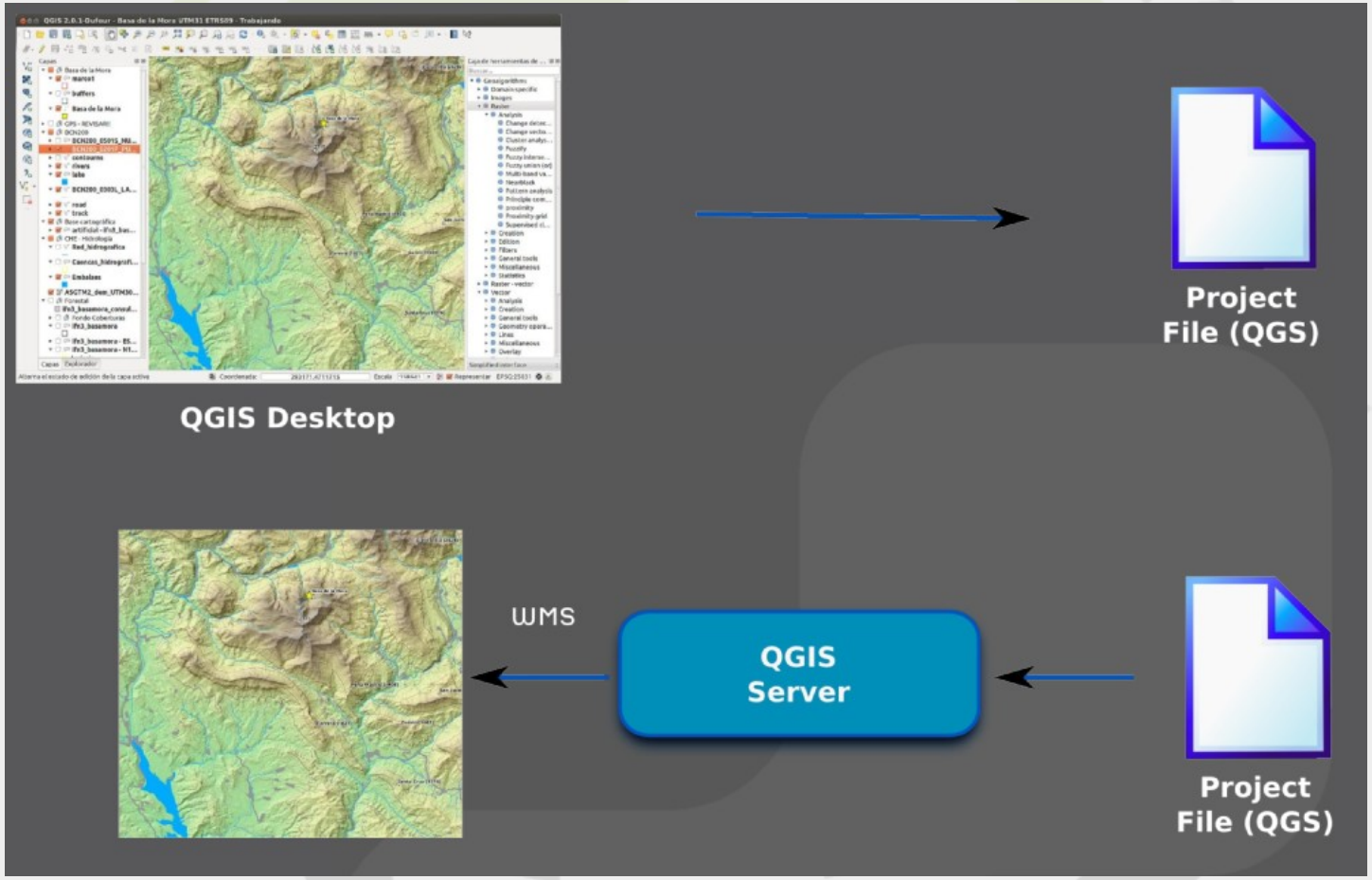
- Annoncé le 11 mai 2007
  - « Dear QGIS developer and users I'm happy to announce the start of the 'QGIS mapserver' project ... The idea of QGIS mapserver is simple: instead of using QGIS just as a desktop GIS, it can also be used as a server. The benefit is that bug fixes and extensions for the server also improve the desktop GIS (and the other way round) ... Contact me if you are interesting in joining development of QGIS mapserver, there is still a lot to do... »

Origines de QGIS Server

- Utiliser QGIS comme un moteur de rendu



QGIS Server


QGIS Desktop
Project File (QGS)
WMS
QGIS Server
Project File (QGS)

QGIS Server

- Partie intégrante de QGIS depuis 2010
- Web Map Service 1.3.0
- Ajout du WMS 1.1.1 en 2012
- Ajout du Web Feature Service 1.0.0 en 2012
- Ajout du Transactional WFS 1.0.0 en 2012
- Ajout du Web Coverage Service 1.0.0 en 2013
- QGIS Mapserver => QGIS Server

QGIS Server

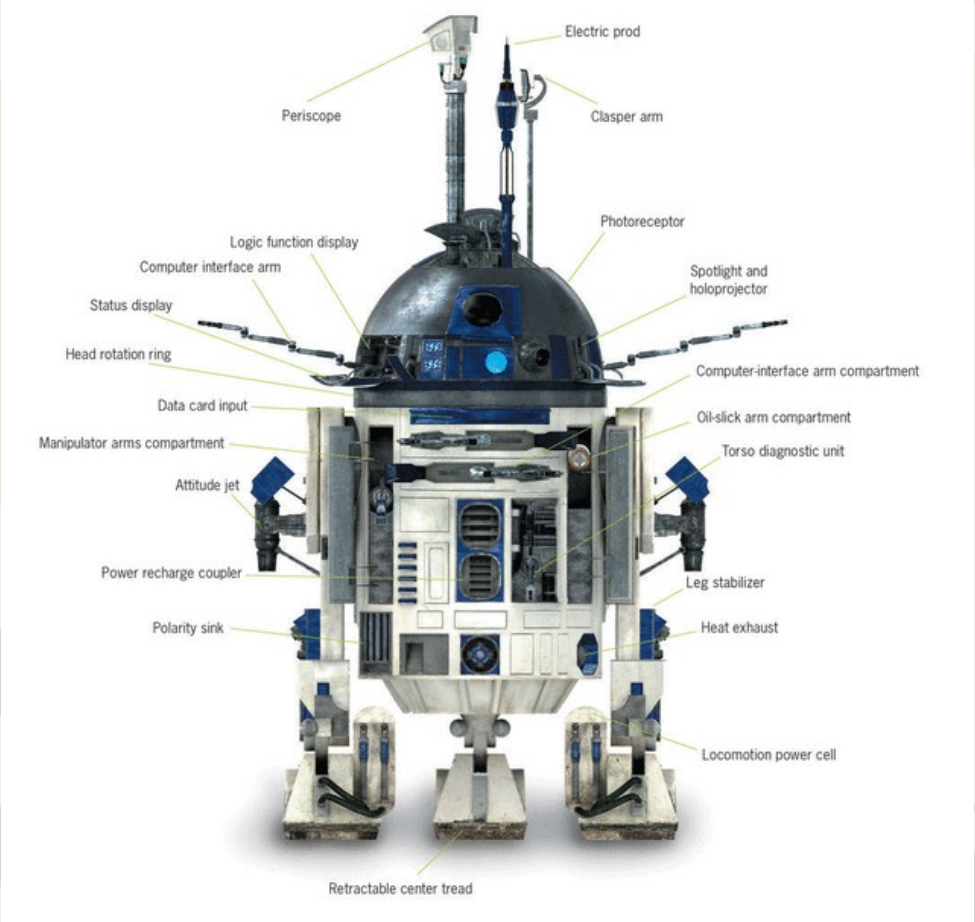
QGIS Server

- Des extensions
  - WMS GetMap au format DXF
  - WMS GetPrint
  - WMS selection, filtre, etc
  - WFS Filtre par expression
  - WFS Simplification des géométries (centroid, extent, none)

Un apport mutuel

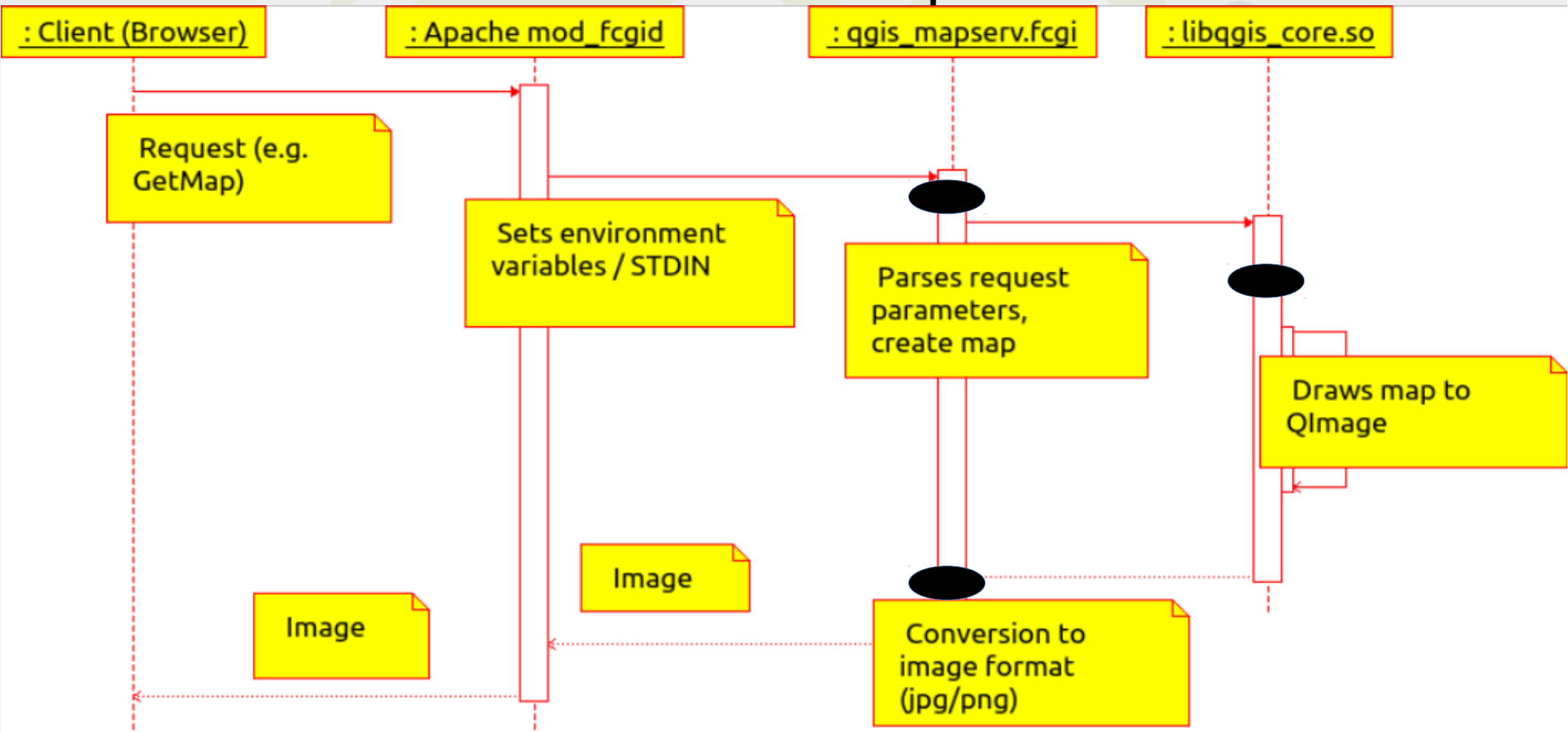
- Heatmap, blend mode, exportToGeoJSON...

Les Extensions Python



Les extensions Python

- Prendre le contrôles des requêtes



Les Extensions Python

- Depuis QGIS 2.8
- Prendre le contrôle des requêtes
  - Modifier les paramètres entrant
    - Forcer un paramètre
  - Modifier la réponse
    - Incrusté un filigrane
  - Contrôler les données
- Ajouter de nouveaux services (standards ou non)

Les Extensions Python

- wmsGetFeatureInfoPrecision
  - Améliorer WMS GetFeatureInfo
  - Ajouter les paramètres
    - FI_POINT_TOLERANCE
    - FI_LINE_TOLERANCE
    - FI_POLYGON_TOLERANCE
  - Depuis QGIS 2.10

Les Extensions Python

- wmsGetFeatureInfoPrecision

__init__.py | wmsGetFeatureInfoPrecision.py

```python
# -*- coding: utf-8 -*-
"""
 This script initializes the plugin, making it known to QGIS.
"""


def serverClassFactory(serverIface):
    from wmsGetFeatureInfoPrecision import ServerGetFeatureInfoPrecision
    return ServerGetFeatureInfoPrecision(serverIface)

def classFactory(iface):
    from wmsGetFeatureInfoPrecision import GetFeatureInfoPrecision
    return GetFeatureInfoPrecision(iface)

```

Les Extensions Python

- wmsGetFeatureInfoPrecision

```
64  class ServerGetFeatureInfoPrecision:
65      """Plugin for QGIS server"""
66
67      def __init__(self, serverIface):
68          # Save reference to the QGIS server interface
69          self.serverIface = serverIface
70          try:
71              self.serverIface.registerFilter(ServerGetFeatureInfoPrecisionFilter(serverIface), 1000)
72          except Exception, e:
73              QgsLogger.debug("ServerGetFeatureInfoPrecision- Error loading filter %s", e)
74
```

Les Extensions Python

```python
FI_POINT_TOLERANCE = 16
FI_LINE_TOLERANCE = 8
FI_POLYGON_TOLERANCE = 4

class ServerGetFeatureInfoPrecisionFilter(QgsServerFilter):

    def requestReady(self):
        request = self.serverInterface().requestHandler()
        params = request.parameterMap( )
        if params.get('SERVICE', '').lower() == 'wms' \
                and params.get('REQUEST', '').lower() == 'getfeatureinfo':
            # Test config file
            if os.path.exists(os.path.join(os.path.dirname(os.path.realpath(__file__)),'config.cfg')):
                config = ConfigParser.ConfigParser()
                config.read(os.path.join(os.path.dirname(os.path.realpath(__file__)),'config.cfg'))

                pointTolerance = config.get('default','FI_POINT_TOLERANCE',str(FI_POINT_TOLERANCE))
                request.setParameter('FI_POINT_TOLERANCE', str(pointTolerance))

                lineTolerance = config.get('default', 'FI_LINE_TOLERANCE', str(FI_LINE_TOLERANCE))
                request.setParameter('FI_LINE_TOLERANCE', str(lineTolerance))

                polygonTolerance = config.get('default', 'FI_POLYGON_TOLERANCE', str(FI_POLYGON_TOLERANCE))
                request.setParameter('FI_POLYGON_TOLERANCE', str(polygonTolerance))
            else:
                request.setParameter('FI_POINT_TOLERANCE', str(FI_POINT_TOLERANCE))
                request.setParameter('FI_LINE_TOLERANCE', str(FI_LINE_TOLERANCE))
                request.setParameter('FI_POLYGON_TOLERANCE', str(FI_POLYGON_TOLERANCE))
```

Les Extensions Python

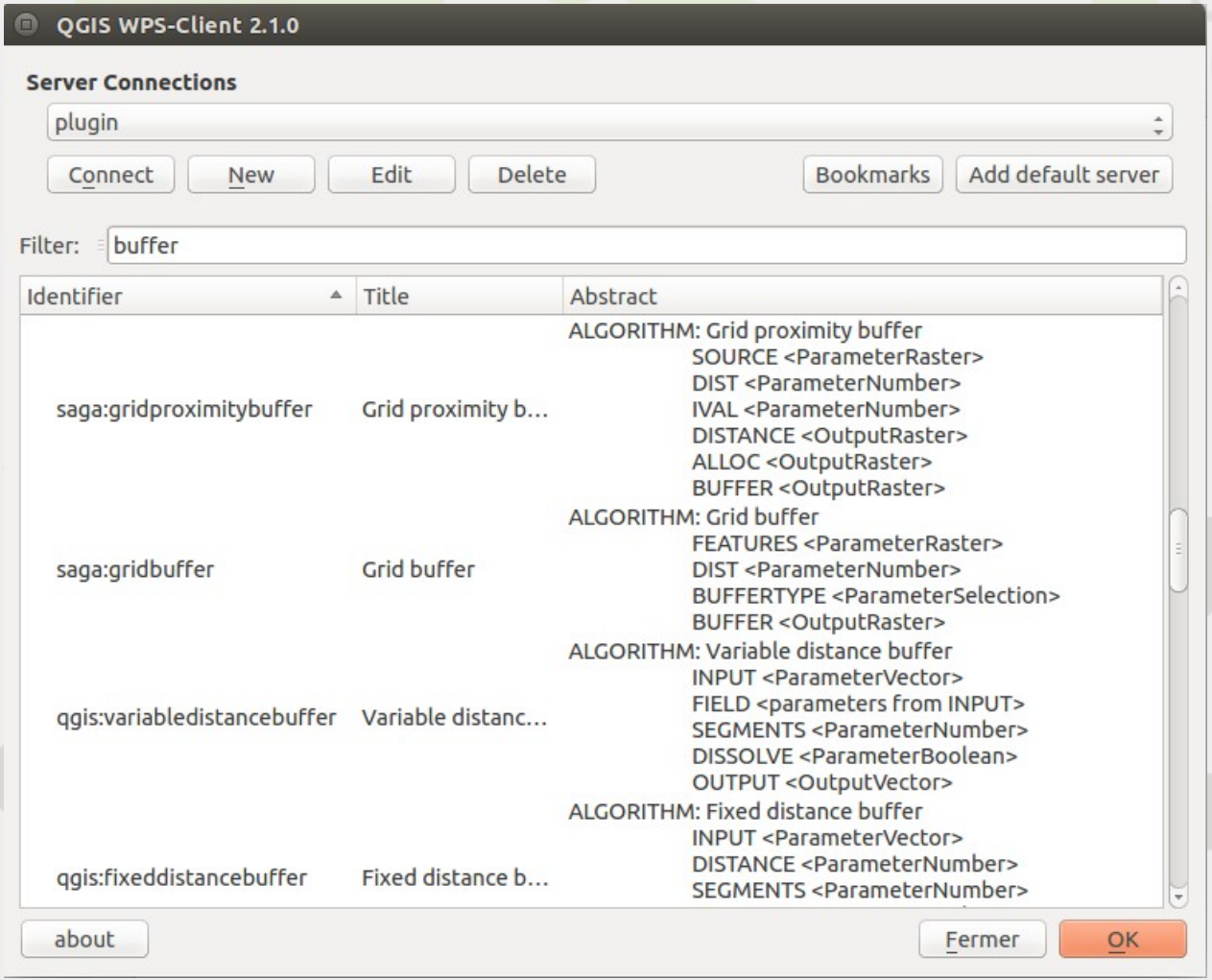
- wmsGetFeatureInfoPrecision

__init__.py | wmsGetFeatureInfoPrecision.py | metadata.txt

```
[general]
name=QGIS Server WMS GetFeatureInfo Precision
qgisMinimumVersion=2.10
qgisMaximumVersion=2.99
description=Set WMS GetFeatureInfo Precision
version=1.0
author=DHONT René-Luc (3Liz)
email=rldhont@3Liz.com
; if True it's a server plugin
server=True

about=wmsGetFeatureInfoPrecision adds precision parameters to WMS GetFeatureInfo Request.

tracker=https://github.com/3liz/qgis-wmsGetFeatureInfoPrecision
repository=https://github.com/3liz/qgis-wmsGetFeatureInfoPrecision
# End of mandatory metadata

# Recommended items:

# Uncomment the following line and add your changelog:
# changelog=

external_deps=none,really

# Tags are comma separated with spaces allowed
tags=server, wms, precision

homepage=https://github.com/3liz/qgis-wmsGetFeatureInfoPrecision
category=server
icon=icon.png
```

Les Extensions Python

- wps4server : Web Processing Service basé sur le module Traitement et pyWPS



Les Extensions Python

- wps4server : Web Processing Service

```
509 class wpsFilter(QgsServerFilter):
510
511     def __init__(self, serverIface):
512         super(wpsFilter, self).__init__(serverIface)
513
514     def requestReady(self):
515         """request ready"""
516         #QgsMessageLog.logMessage("wpsFilter.requestReady")
517
518
519     def sendResponse(self):
520         """send response"""
521         #QgsMessageLog.logMessage("wpsFilter.sendResponse")
522
523     def responseComplete(self):
524         QgsMessageLog.logMessage("wpsFilter.responseComplete")
525         request = self.serverInterface().requestHandler()
526         params = request.parameterMap()
527         service = params.get('SERVICE', '')
528         if service and service.upper() == 'WPS':
529             # prepare query
530             inputQuery = '&'.join(["%s=%s" % (k, params[k]) for k in params if k.lower() != 'm
531             request_body = params.get('REQUEST_BODY', '')
532
```

Les Extensions Python

- wfsOutputExtension

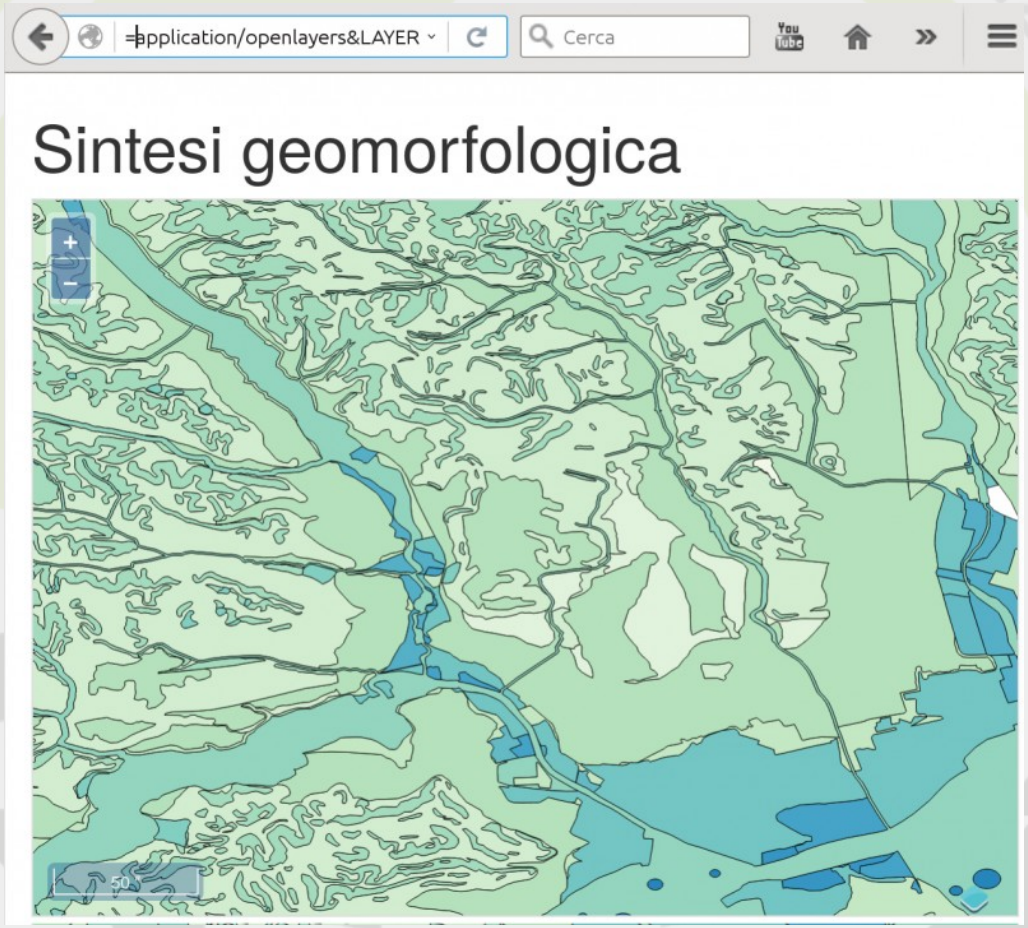
```
−<GetFeature>
  −<ResultFormat>
      <GML2/>
      <GML3/>
      <GeoJSON/>
      <SHP/>
      <XLSX/>
      <ODS/>
      <KML/>
      <MIF/>
      <TAB/>
      <CSV/>
    </ResultFormat>
  −<DCPType>
```

Les Extensions Python

- wfsOutputExtension

```
94   class WFSFilter(QgsServerFilter):
95
96       def __init__(self, serverIface):
97           QgsMessageLog.logMessage("WFSFilter.init")
98           super(WFSFilter, self).__init__(serverIface)
99           self.format = None
100          self.typename = ""
101          self.filename = ""
102
103          self.tempdir = os.path.join( tempfile.gettempdir(), 'qgis_wfs' )
104          if not os.path.exists(self.tempdir):
105              os.mkdir( self.tempdir )
106          QgsMessageLog.logMessage("WFSFilter.tempdir: %s" % self.tempdir)
107
108      def requestReady(self):
130
131      def sendResponse(self):
191
192      def responseComplete(self):
213
```

Les Extensions Python

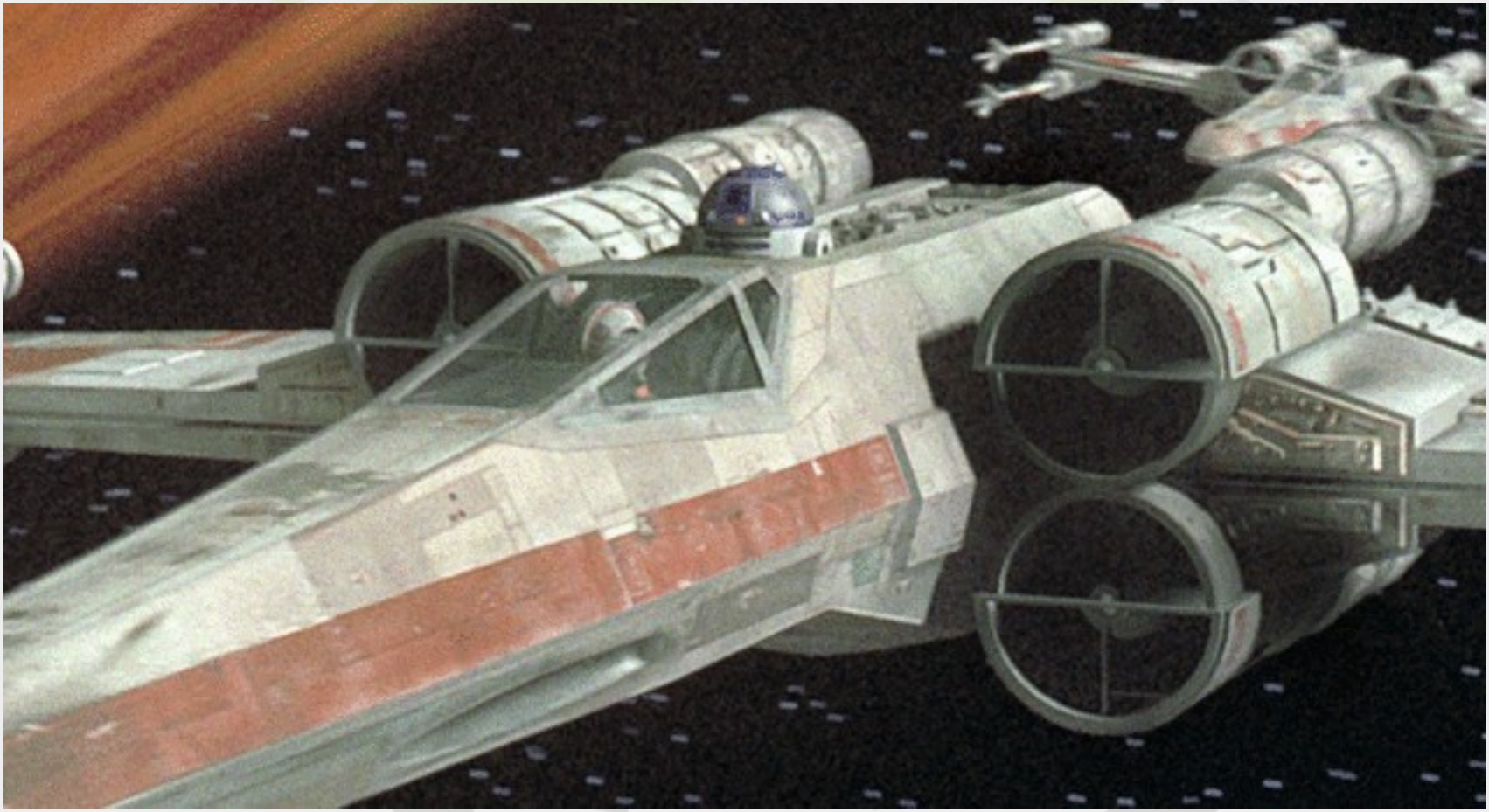


Les extensions Python

- Contrôle de l'accès aux données

```
82  class RestrictedAccessControl(QgsAccessControlFilter):
83
84      """ Used to have restriction access """
85
86      # Be able to deactivate the access control to have a reference point
87      _active = False
88
89      def __init__(self, server_iface):
91
92      def layerFilterExpression(self, layer):
99
100     def layerFilterSubsetString(self, layer):
114
115     def layerPermissions(self, layer):
135
136     def authorizedLayerAttributes(self, layer, attributes):
145
146     def allowToEdit(self, layer, feature):
153
154     def cacheKey(self):
156
157
158 server = QgsServer()
159 server.handleRequest()
160 server_iface = server.serverInterface()
161 accesscontrol = RestrictedAccessControl(server_iface)
162 server_iface.registerAccessControl(accesscontrol, 100)
163
```

L'API qgis.server



L'API qgis.server

- Depuis QGIS 2.12
- Faciliter la création de tests
- Embarquer QGIS Server

L'API qgis.server

test_qgsserver.py

```python
# -*- coding: utf-8 -*-
"""QGIS Unit tests for QgsServer.

.. note:: This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.
"""
__author__ = 'Alessandro Pasotti'
__date__ = '25/05/2015'
__copyright__ = 'Copyright 2015, The QGIS Project'
# This will get replaced with a git SHA1 when you do a git archive
__revision__ = '$Format:%H$'

import os
import re
import unittest
import urllib
from qgis.server import QgsServer
from qgis.core import QgsMessageLog
from utilities import unitTestDataPath

# Strip path and content length because path may vary
RE_STRIP_PATH = r'MAP=[^&]+|Content-Length: \d+'


class TestQgsServer(unittest.TestCase):
```

L'API qgis.server

```
82  class RestrictedAccessControl(QgsAccessControlFilter):
83
84      """ Used to have restriction access """
85
86      # Be able to deactivate the access control to have a reference point
87      _active = False
88
89      def __init__(self, server_iface):
91
92      def layerFilterExpression(self, layer):
99
100     def layerFilterSubsetString(self, layer):
114
115     def layerPermissions(self, layer):
135
136     def authorizedLayerAttributes(self, layer, attributes):
145
146     def allowToEdit(self, layer, feature):
153
154     def cacheKey(self):
156
157
158 server = QgsServer()
159 server.handleRequest()
160 server_iface = server.serverInterface()
161 accesscontrol = RestrictedAccessControl(server_iface)
162 server_iface.registerAccessControl(accesscontrol, 100)
163
```

L'API qgis.server

```
144     def wms_request_compare(self, request, extra=None, reference_file=None):
145         project = self.testdata_path + "test+project.qgs"
146         assert os.path.exists(project), "Project file not found: " + project
147 
148         query_string = 'MAP=%s&SERVICE=WMS&VERSION=1.3&REQUEST=%s' % (urllib.quote(project), request)
149         if extra is not None:
150             query_string += extra
151         header, body = [str(_v) for _v in self.server.handleRequest(query_string)]
152         response = header + body
153         f = open(self.testdata_path + (request.lower() if not reference_file else reference_file) + '.txt')
154         expected = f.read()
155         f.close()
156         # Store the output for debug or to regenerate the reference documents:
157         """
158         f = open(os.path.dirname(__file__) + '/expected.txt', 'w+')
159         f.write(expected)
160         f.close()
161         f = open(os.path.dirname(__file__) + '/response.txt', 'w+')
162         f.write(response)
163         f.close()
164         """
165         response = re.sub(RE_STRIP_PATH, '', response)
166         expected = re.sub(RE_STRIP_PATH, '', expected)
167 
168         # for older GDAL versions (<2.0), id field will be integer type
169         if int(osgeo.gdal.VersionInfo()[:1]) < 2:
170             expected = expected.replace('typeName="Integer64" precision="0" length="10" editType="TextEdit" typ
171 
172         self.assertEqual(response, expected, msg="request %s failed.\n Query: %s\n Expected:\n%s\n\n Response:\
173 
```

L'API qgis.server

- python/server/qgsserverinterface.sip
- python/server/qgsserverfilter.sip
- python/server/qgsaccesscontrolfilter.sip

- tests/src/python/test_qgsserver.py
- tests/src/python/test_qgsserver_accesscontrol.py

L'API qgis.server

```
3.  Super simple QgsServer.
4.  """
5.
6.  from qgis.server import *
7.  from BaseHTTPServer import *
8.
9.  class handler (BaseHTTPRequestHandler):
10.
11.     server = QgsServer()
12.
13.     def _doHeaders(self, response):
14.         l = response.pop(0)
15.         while l:
16.             h = l.split(':')
17.             self.send_header(h[0], ':'.join(h[1:]))
18.             self.log_message( "send_header %s - %s" % (h[0], ':'.join(h[1:])))
19.             l = response.pop(0)
20.         self.end_headers()
21.
22.     def do_HEAD(self):
23.         self.send_response(200)
24.         response = str(handler.server.handleRequestGetHeaders(self.path[2:])).split(
25.         self._doHeaders(response)
26.
27.     def do_GET(self):
28.         response = str(handler.server.handleRequest(self.path[2:])).split('\n')
29.         i = 0
30.         self.send_response(200)
31.         self._doHeaders(response)
32.         self.wfile.write(('\n'.join(response[i:])).strip())
33.
34.     def do_OPTIONS(s):
35.         handler.do_GET(s)
36.
37. httpd = HTTPServer( ('', 8000), handler)
```

L'API qgis server

```python
# QGIS server view

from django.http import HttpResponse
from django.views.generic import View
from qgis.server import *


class OGC(View):
    """Pass a GET request to QGIS Server and return the response"""

    def __init__(self):
        self.server = QgsServer()

    def get(self, request, *args, **kwargs):
        """Pass a GET request to QGIS Server and return the response"""
        headers, body = self.server.handleRequest(request.GET.urlencode())
        response = HttpResponse(body)
        # Parse headers
        for header in headers.split('\n'):
            if header:
                k, v = header.split(': ', 1)
                response[k] = v
        return response

```

L'API qgis.server

Tornado

L'avenir de QGIS Server

L'avenir de QGIS Server

- ~~Meilleur respect de la norme ISO~~
- ~~Faciliter la saisie des propriétés~~
- Plugin Server WMTS
- ~~Outil Bureautique de validation~~
- ~~ShowFeatureCount pour GetLegendGraphic~~
- ~~WMS INSPIRE~~
- WFS 2 ?

FOSS4G-fr 2016

Merci !

Des Questions ?