camp
tocamp

INNOVATIVE SOLUTIONS
BY OPEN SOURCE EXPERTS

GraphQL API on Odoo
Powered by OCA

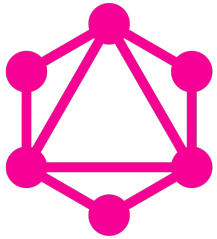
Yannick Payot

Twitter/Github: yvaucher

Agenda

camptocamp

- Why GraphQL
  - What is it?
  - Project context
- Sneak peek in the graphical interface
- How to develop your own API
  - /graphql & /graphiql
  - Authentication
  - Schema / Object definition
  - Resolvers



GraphQL

Ask what you need,

Get exactly that

```
{
  allRoutes(limit: 2, offset: 2) {
    totalCount
    edges {
      node {
        _id
        changeDate
        title
      }
    }
  }
}
```


```
{
  "data": {
    "allRoutes": {
      "totalCount": 2,
      "edges": [
        {
          "node": {
            "_id": 817,
            "changeDate": "2022-07-21T15:49:27.336890",
            "title": {
              "de": "Alpenpanorama-Weg",
              "it": "Sentiero alpino panoramico ",
              "en": "Alpine Panorama Trail",
              "fr": "Chemin panorama alpin"
            }
          }
        },
        {
          "node": {
            "_id": 829,
            "changeDate": "2022-09-22T12:49:57.084701",
```

OCA Module

- graphql_base
  - https://github.com/oca/rest-framework
  - By Acsone
  - Based on  Python-graphene

SwitzerlandMobility

A network for non-motorized traffic

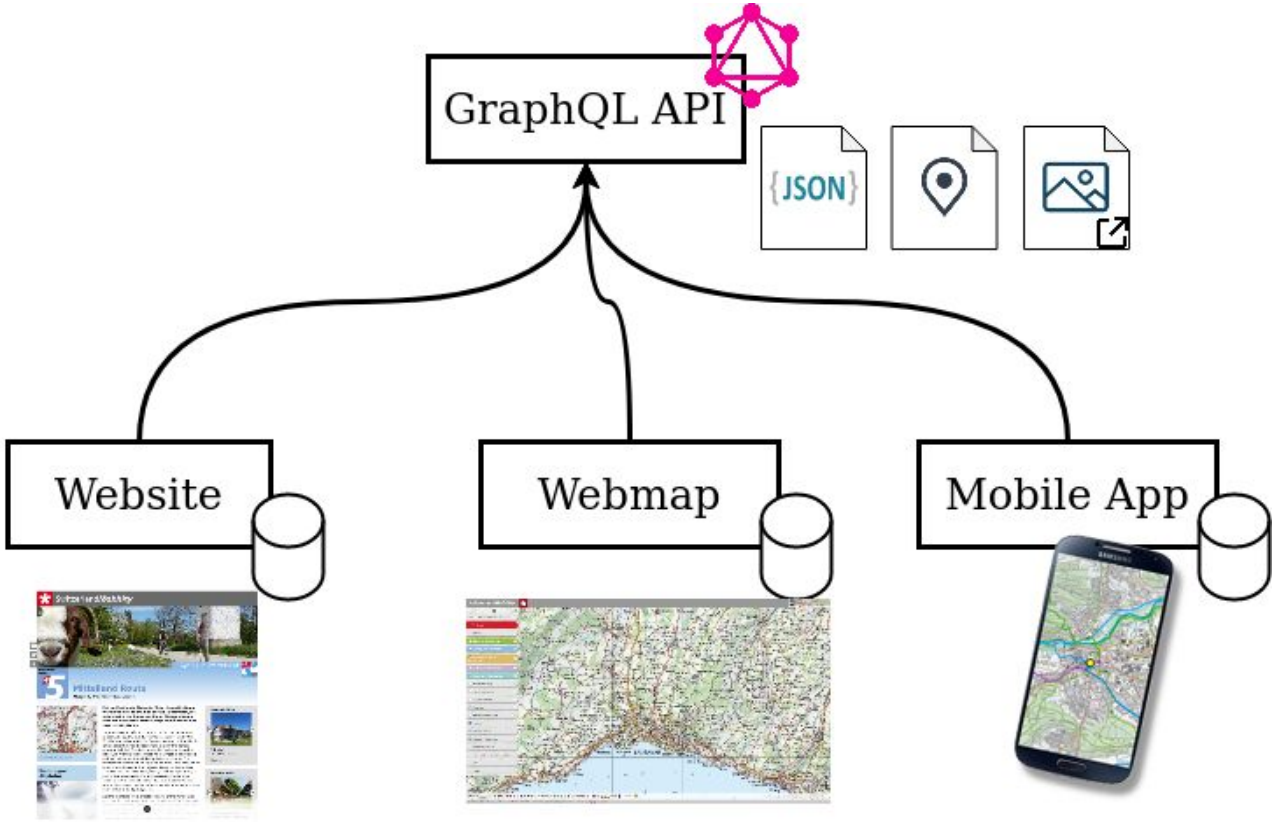
SwitzerlandMobility

camptocamp

- Odoo as a framework
  - geoengine
  - ~ 40 Custom objects
  - 55 users/editors in 40 companies
- Workflow (ETL like)
  - Import
  - Edit
  - Validate
  - Publish (revision system)
- API
  - Rest API → GraphQL API
  - 3 data consumers

SwitzerlandMobility

camptocamp



Let’s have a look

Test your endpoint

/graphiql

- → /graphql
  - API Endpoint
- → /graphiql
  - Graphic interface
    - Auto-completion
    - Auto-documented

Authentication

OCA/server-auth

auth_api_key

```
@http.route("/graphql", auth="api_key", csrf=False)
def graphql(self, **kwargs):
        return self._handle_graphql_request(schema)
```

```
curl "http://myodoo.com/graphql" \
   -d "query={allRoutes {edges {node {geom}}}}" \
   -H "Accept: application/json" \
   -H "API-Key: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
```

Schema

Register the available queries

```python
from graphene import Field, Int, ObjectType, Schema, String, relay

from .land import Land
from .route import Route, RouteConnection

class Query(ObjectType):
    land = Field(Land, code=String(required=True, description=LAND_
    route = Field(
        Route,
        land=String(required=True, description=LAND_DESCR),
        number=Int(required=True),
        description="Main object, can be composed of Segments",
    )
    all_routes = relay.ConnectionField(
        RouteConnection,
        land=String(description=LAND_DESCR),
        since=DateTime(),
        limit=Int(),
        offset=Int(),
        description="List of the main object, can be composed of Se
    )

    @staticmethod
    def resolve_route(root, info, number=0, land=None):
        ...

    @staticmethod
    def resolve_all_routes(root, info, land=None, since=None, **kwa
       ...

schema = Schema(query=Query)
```

Object definition

How to map your Odoo objects

```python
# Copyright 2021 Camptocamp SA
# License AGPL-3.0 or later (http://www.gnu.org/licenses/agpl).
from graphene import List, String
from odoo.addons.graphql_base import OdooObjectType

from . import resolver
from .common import WebDB
from .land import Land
from .tools import get_public_url
from .types import Translation

# pylint: disable=no-self-argument


class File(OdooObjectType):
    class Meta:
        interfaces = (WebDB,)

    name = String(resolver=WebDB.resolve_name)

    lands = List(Land, resolver=resolver.x2m)
    title = Translation()
    url = String()

    def resolve_url(root, info):
        return get_public_url(root.file_url)
```

Object definition - Cycling reference

camptocamp

- ➔ Graphene / GraphQL limitation
- ➔ Cycling reference
  - Papermaps → Route
  - Route → Papermaps
- ➔ Deferred import
  - def getRoute():
    from .route import Route
    return Route
  - class PaperMap(OdooObjectType):
    routes = List(getRoute, ...)

Field names

CamelCase or snake_case

➔ GraphQL

- Translated to CamelCase
- Data consumer don’t need to know about Odoo
- Can be switched off

Resolvers

camptocamp

- Implicit default resolver
  - root.name
    - name = String()
- Field type resolver
  - Default resolver on type
    - name = Translation()
- Specific field resolver
  - By field name
    - name = String()
      def resolver_name(...):
  - Or field attribute
    - name = String(resolver=myresolver)

Resolver - Fetching

camptocamp

- `root` contains the Odoo env
- root.env[“res.country”].search(domain)

Resolvers - tips

camptocamp

- ➔ Odoo agnostic
  - Relations, remove the _id, _ids suffixes
  - False → null

Resolvers - Examples

camptocamp

- Translations (New type)
  - 4 languages json
- Geometries / Coordinates (New type)
  - Complex geometries url
    - Url to geojson / geobuff endpoint
  - Coordinates
    - [x, y]
- Relations (Field attribute)
  - Ordered M2M

Questions


camptocamp
INNOVATIVE SOLUTIONS
BY OPEN SOURCE EXPERTS