atmantree.com

El autor del presente documento lo ha publicado bajo las condiciones que especifica la licencia

Creative Commons
Attribution-NonCommercial-ShareAlike 3.0

http://creativecommons.org/licenses/by-nc-sa/3.0/

En caso de dudas escriba a:
info@atmantree.com

Web Mapping con Django

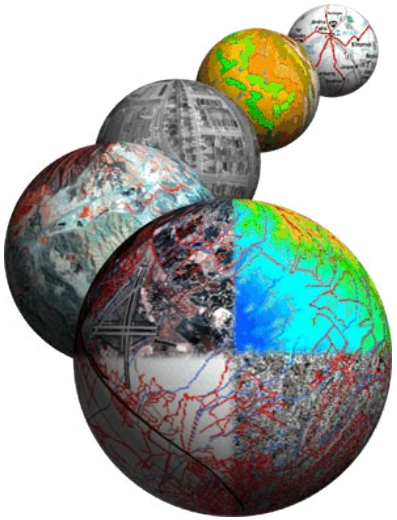
Ing. Carlos Gustavo Ruiz
http://atmantree.com
Noviembre, 2012

CC BY-NC-SA 3.0

Temas

- ¿Qué es Geomática?
- ¿Qué es Web Mapping?
- Paso a paso
  - Instalación
  - Base de Datos
  - Proyecto Django
  - Modelo (Geo)
  - GeoAdmin
  - Plantilla
  - Vista
  - OpenLayers
  - Nuestro Portal
- Lo que sigue
- A modo de Conclusión

¿Qué es Geomática?



[Geo] + Infor**[mática]** = **[Geomática]**

¿Qué es Geomática?

Palabras clave:

- Conjunto de ciencias
- Integración
- Medios para la captura
- Tratamiento
- Análisis
- Interpretación
- Difusión
- Almacenamiento
- Información Geográfica.

¿Qué es Geomática Libre?

Entendemos que la Geomática Libre es en cierto modo un superconjunto del Software Libre pues integra las 4 libertades de Uso, Introspección, Mejora y Copias a aspectos como:

- Software
- Datos
- Estándares

¿Qué es Web Mapping?

Es el proceso de diseñar, implementar, generar y despachar mapas en la World Wide Web y sus productos.

Pueden ser:

- Analíticos (generalmente producto de un GIS)
- Animados (para mostrar eventos como cursos de agua, comunicaciones, flujos, estudios sociales)
- Colaborativos (OSM, Google, Wikimapia)
- Dinámicos (creados a demanda del usuario)
- Atlas en línea
- Mapas en Tiempo Real (mapas de clima, tráfico, tracking)
- Mapas estáticos (generalmente imágenes y fotos)

http://en.wikipedia.org/wiki/Web_mapping

Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para el Ambiente | Instituto Geográfico de Venezuela Simón Bolívar

GEOPORTAL SIMÓN BOLÍVAR

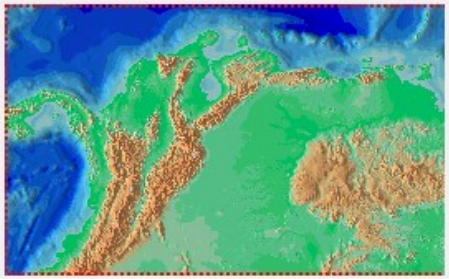
Herramientas

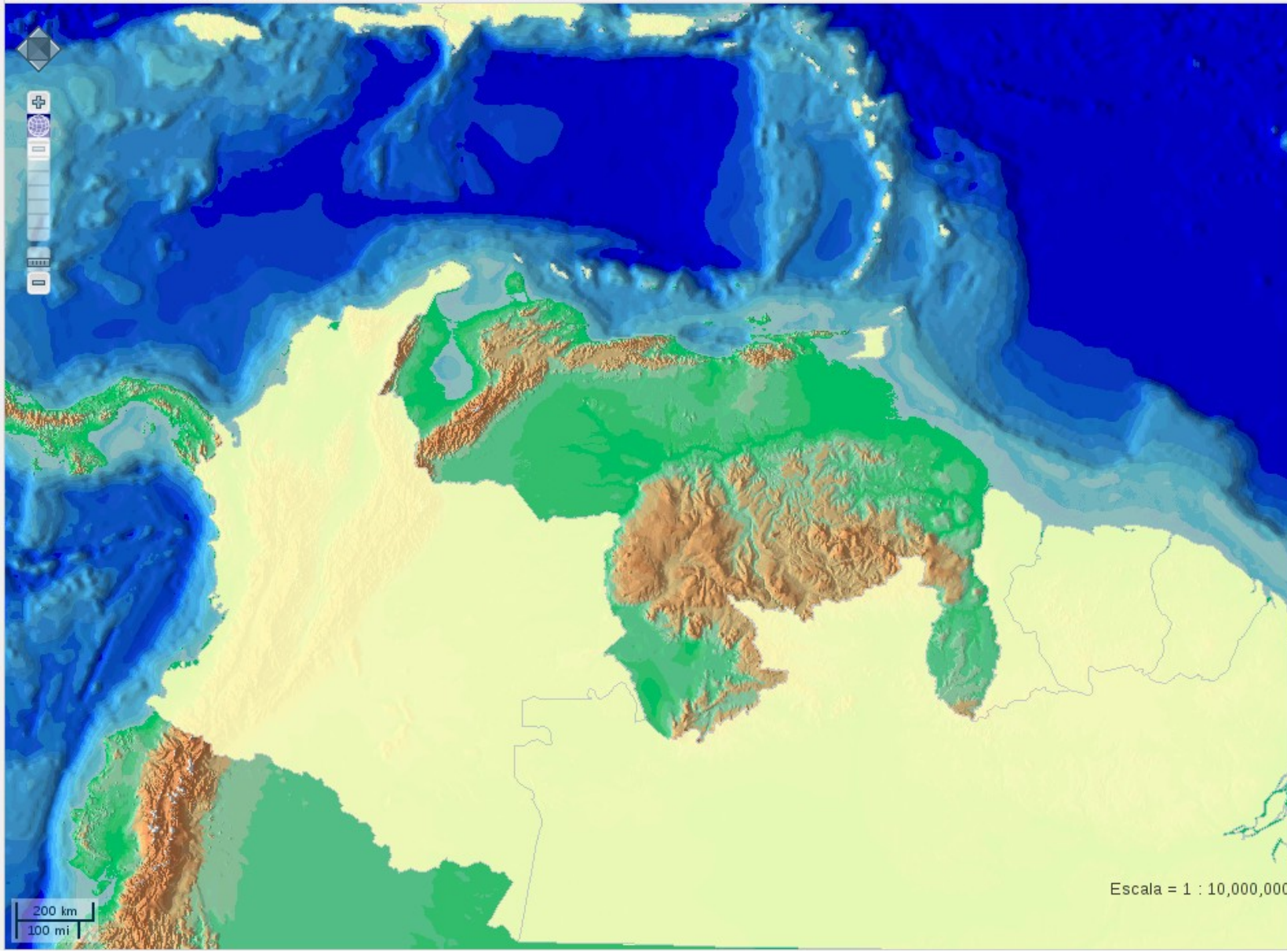
Información ampliada

Capas

- Capitales de Estados
- Áreas Urbanas
- Troncales
- Cuerpos de Aguas
- Hidrografía
- Curvas de Nivel
- División Municipal
- División Estadal
- Territorio Nacional
- Límites Marítimos
- Paises Vecinos
- Comunas

Vista general



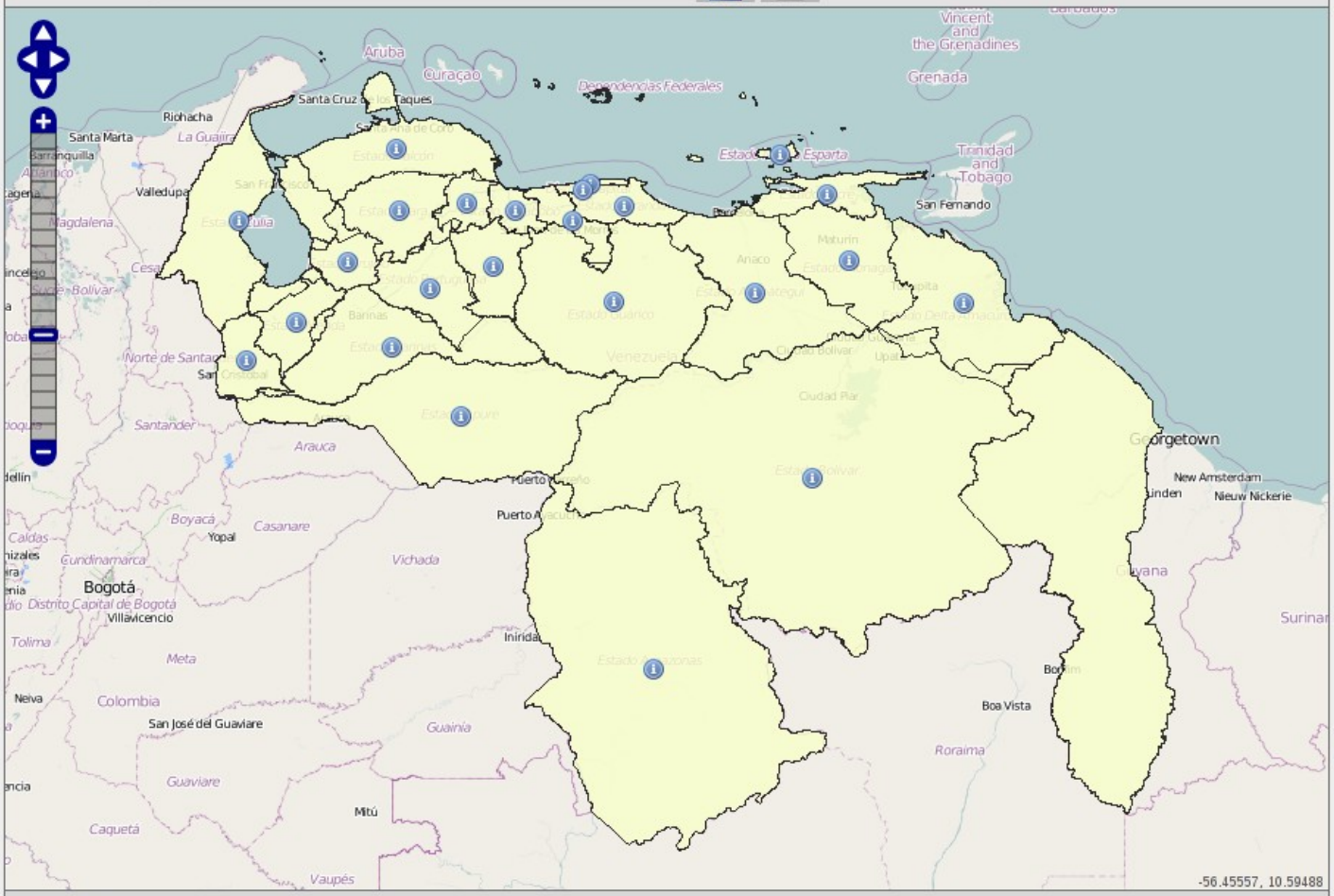


Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para Ciencia, Tecnología e Innovación

Seleccione una Zona de Interés | Información

Capas de Información: ☑ Información del Portal | ☐ Capitales de Estados | ☐ Capitales de Municipio | ☐ Municipios | ☑ Estados

Buscar lugares:



Ayuda | Políticas de Uso | Acerca de

Portal desarrollado y administrado por el Centro Nacional de Tecnologías de Información, adscrito al Ministerio del Poder Popular para Ciencia, Tecnología e Innovación.

Seleccione una Zona

Oficinas Postales

Zonas Postales

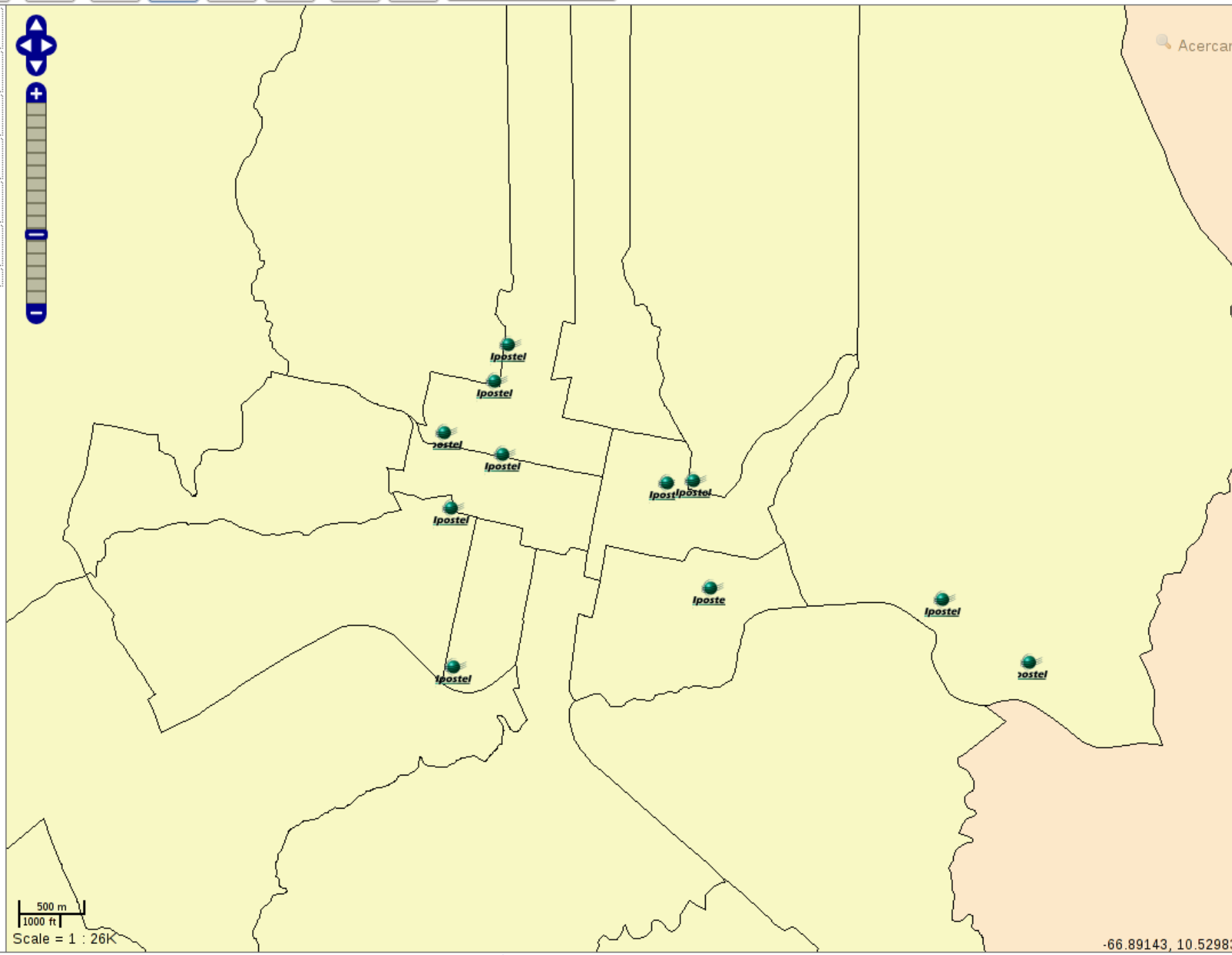
Manzanas

Parroquias

Municipios

Estados

Regiones (Capa Base)



Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para Ciencia, Tecnología e Innovación | Red de Transmisiones de Venezuela, C.A. (REDTV)

200 BICENTENARIO

Seleccione una Zona

- Sedes Regionales
- Cobertura VTV
- Cobertura Vive
- Cobertura TVes
- Cobertura Telesur
- Cobertura RNV Juvenil
- Cobertura RNV Informativa
- Cobertura RNV Indigena
- Cobertura RNV Clasica
- Regiones

Cobertura VTV	
Región:	CENTRO
Estado:	COJEDES
Municipio:	SAN CARLOS
Parroquia:	MANUEL MANRIQUE
Estación:	LA GLORIA (SAN CARLOS)
Potencia:	2100
Habitantes:	0.00000000000

Cobertura VTV	
Región:	CENTRO
Estado:	CARABOBO
Municipio:	CARLOS ARVELO
Parroquia:	TACARIGUA
Estación:	ALETON
Potencia:	20000
Habitantes:	0.00000000000

Cobertura Vive	
Región:	CENTRO
Estado:	COJEDES
Municipio:	SAN CARLOS
Parroquia:	MANUEL MANRIQUE
Estación:	LA GLORIA (SAN CARLOS)
Potencia:	
Habitantes:	0.00000000000

Cobertura Vive	
Región:	CENTRO
Estado:	CARABOBO
Municipio:	CARLOS ARVELO
Parroquia:	TACARIGUA
Estación:	ALETÓN
Potencia:	
Habitantes:	0.00000000000

Cobertura Vive

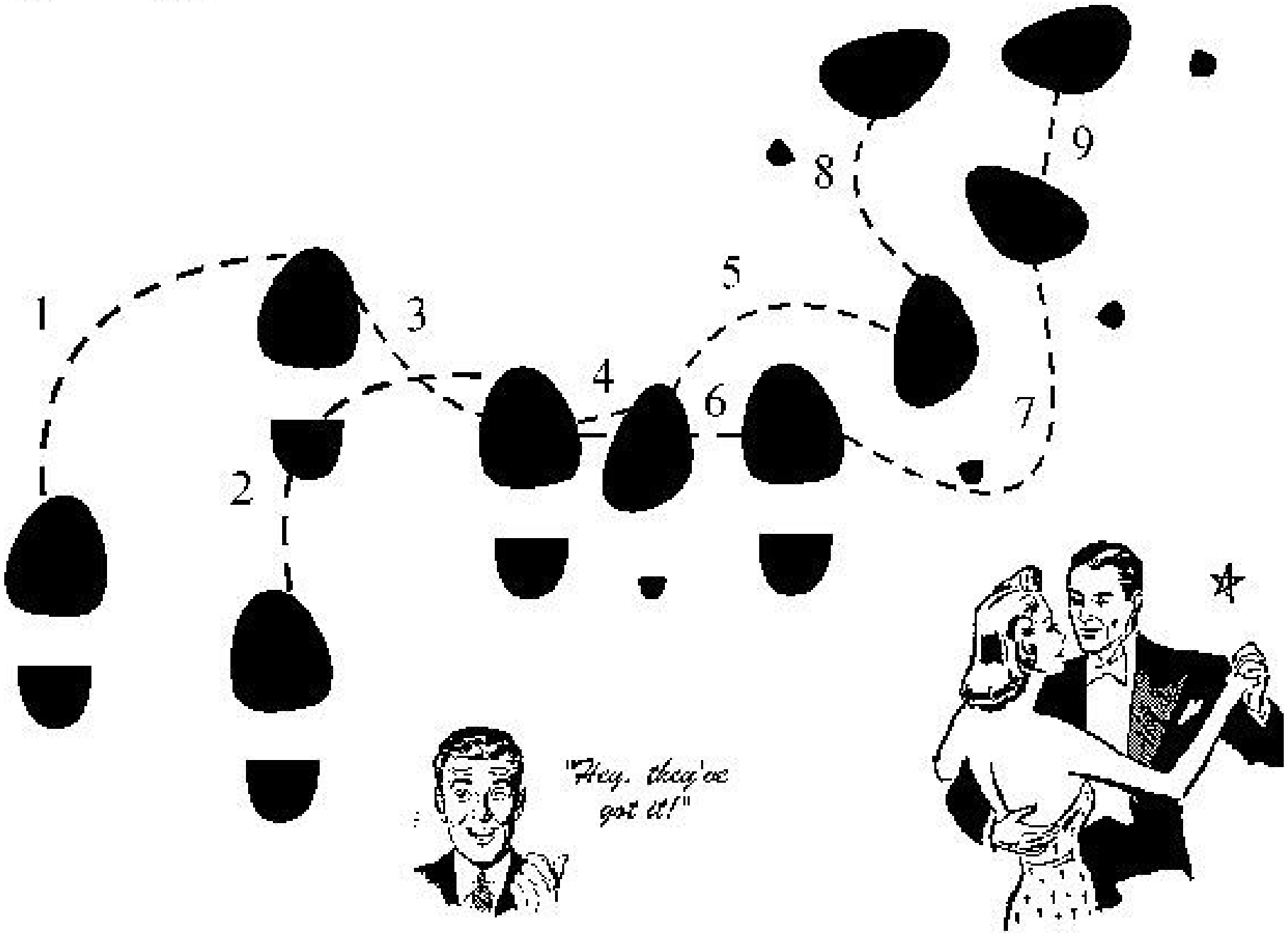
100 km
50 mi
Scale = 1 : 3M

64.09226, 9.7478

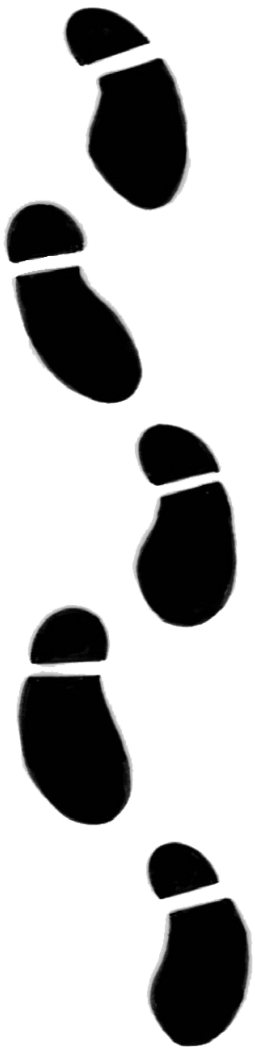
Créditos

Vamos con el
Paso a Paso

Rhumba!



Paso a Paso



- Instalación
- Base de Datos
- Proyecto Django
- Modelo (Geo)
- GeoAdmin
- Plantillas
- Vista
- OpenLayers
- Portal

Paso a Paso

- Instalación
  - En Debian y derivados

```
$ su
# aptitude install postgresql postgresql-client postgresql-server-dev-all
# aptitude install postgresql-8.4-postgis postgis
# aptitude install python-virtualenv virtualenvwrapper
# exit
$ mkdirvirtualenv gdj
......
(gdj) $ pip install Django psycopg2
....
(gdj) $
```

  - No cierre esta consola

Paso a Paso

- Base de Datos
  - pg_hba.conf (en /etc/postgresql/8.4/main/) (solo para efectos del taller)

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD

# "local" is for Unix domain socket connections only
local   all         all                                     trust
# IPv4 local connections:
host    all         all         127.0.0.1/32          trust
# IPv6 local connections:
host    all         all         ::1/128                   md5
```

  - (Luego reiniciar PostgreSQL)

Paso a Paso

- Base de Datos
  - Crear la base de datos e instalar esquema PostGIS

```
$ su -
# su posrgres
# createdb -E utf-8 geodb
# createlang plpgsql geodb
# psql -f /usr/share/postgresql/8.4/contrib/postgis-1.5/postgis.sql geodb
# psql -f /usr/share/postgresql/8.4/contrib/postgis-1.5/spatial_ref_sys.sql geodb
# exit
# exit
$
```

Paso a Paso

- Proyecto Django
  - En la consola inicial

```
(gdj) $ django-admin.py startproject web_mapping
(gdj) $ cd web_mapping/
(gdj) $ ls
manage.py  web_mapping
(gdj) $ nano web_mapping/settings.py          # ver siguiente lámina
```

Paso a Paso

- Proyecto Django
  - En la consola inicial

```
DATABASES = {
    'default': {
        'ENGINE': 'django.contrib.gis.db.backends.postgis',
        'NAME': 'geodb',
        'USER': 'postgres',                    #sólo a efectos del taller
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}

INSTALLED_APPS = (
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.sites',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'django.contrib.admin',
    'django.contrib.gis',
    'app1',
)
```

Paso a Paso

- Proyecto Django
  - En la consola inicial

```
(gdj) $ django-admin.py startproject web_mapping
(gdj) $ cd web_mapping/
(gdj) $ ls
manage.py  web_mapping
(gdj) $ nano web_mapping/settings.py          # ver siguiente lámina
(gdj) $ python manage.py startapp app1
(gdj) $ cd app1/
__init__.py  models.py  tests.py  views.py
(gdj) $ nano models.py                        #ver siguiente lámina
.....
```

Paso a Paso

- Modelo (Geo)

```
from django.contrib.gis.db import models

# Este es un layer en su mapa para guardar los Puntos De Interes de su
# Aplicacion.

class PuntosDeInteres(models.Model):
    nombre = models.CharField(max_length=50)
    descripcion = models.TextField(blank=True, null=True)

    # Aqui es donde GeoDjango hace su magia negra
    pto = models.PointField()
    objects = models.GeoManager()

    def __unicode__(self):
        return self.nombre

    class Meta():
         verbose_name = "Punto de Interes"
         verbose_name_plural = "Puntos de Interes"
```

Paso a Paso

- Modelo (Geo)

```
(gdj) $ cd ..
(gdj) $ python manage.py validate
0 errors found
(gdj) $ python manage.py syncdb
Creating tables ...
Creating table auth_permission
Creating table auth_group_permissions
Creating table auth_group
Creating table auth_user_user_permissions
Creating table auth_user_groups
Creating table auth_user
Creating table django_content_type
Creating table django_session
Creating table django_site
Creating table django_admin_log
Creating table app1_puntosdeinteres
```

Paso a Paso

- Modelo (Geo)

```
You just installed Django's auth system, which means you don't have any
superusers defined.
Would you like to create one now? (yes/no): yes
Username (leave blank to use 'atmantree'): admin
E-mail address: admin@adm.in
Password:
Password (again):
Superuser created successfully.
Installing custom SQL ...
Installing indexes ...
Installed 0 object(s) from 0 fixture(s)
(gdj) $
```

Paso a Paso

- Modelo (Geo)

```
(gdj) $ python manage.py sql app1
BEGIN;
CREATE TABLE "app1_puntosdeinteres" (
    "id" serial NOT NULL PRIMARY KEY,
    "nombre" varchar(50) NOT NULL,
    "descripcion" text
)
;
COMMIT;
(gdj) $ python manage.py dbshell
psql (8.4.13)
conexión SSL (cifrado: DHE-RSA-AES256-SHA, bits: 256)
Digite «help» para obtener ayuda.

geodb=# \d app1_puntosdeinteres
```

Paso a Paso

- Modelo (Geo)

```
                                    Tabla «public.app1_puntosdeinteres»
   Columna   |         Tipo          |                          Modificadores
-------------+-----------------------+-----------------------------------------------------------------------
 id                | integer               | not null valor por omisión nextval('app1_puntosdeinteres_id_seq'::regclass)
 nombre        | character varying(50) | not null
 descripcion  | text                  |
 pto         | geometry              | not null
Índices:
    "app1_puntosdeinteres_pkey" PRIMARY KEY, btree (id)
    "app1_puntosdeinteres_pto_id" gist (pto)
Restricciones CHECK:
    "enforce_dims_pto" CHECK (st_ndims(pto) = 2)
    "enforce_geotype_pto" CHECK (geometrytype(pto) = 'POINT'::text OR pto IS NULL)
    "enforce_srid_pto" CHECK (st_srid(pto) = 4326)

geodb=#
```

Paso a Paso

- Modelo (Geo)

```
geodb=# \q
(gdj) $
```

  - y con esto ya tenemos nuestro modelo geográfico instalado en PostgreSQL. GeoDjango funciona con otros backends de base de datos, sin embargo PostGIS (PostgreSQL) es quien tiene mayor cantidad de funcionalidades implemementadas.

Paso a Paso

- GeoAdmin

```
(gdj) $ nano app1/admin.py
....
from django.contrib.gis import admin
from models import PuntosDeInteres

admin.site.register(PuntosDeInteres, admin.OSMGeoAdmin)
....
(gdj) $ nano web_mapping/urls.py
....
from django.conf.urls import patterns, url, include
from django.contrib.gis import admin

admin.autodiscover()

urlpatterns = patterns('',
    url(r'^admin/', include(admin.site.urls)),
)
....
(gdj) $ python manage.py runserver
```

Paso a Paso

- GeoAdmin

Page not found (404)

Request Method: GET
Request URL: http://localhost:8000/

Using the URLconf defined in `web_mapping.urls`, Django tried these URL patterns, in this order:

1. `^admin/`

The current URL, , didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

Paso a Paso

- GeoAdmin

Paso a Paso

- GeoAdmin

Administración de Django

Bienvenido/a, **admin**. Cambiar contraseña / Terminar sesión

Inicio › App1 › Puntos de Interes › Añadir Punto de Interes

Añadir Punto de Interes

Nombre:

Descripcion:

Pto:

Delete all Features

Grabar y añadir otro | Grabar y continuar editando | **Grabar**

Administración de Django

Bienvenido/a, **admin**. Cambiar contraseña / Terminar sesión

Inicio › App1 › Puntos de Interes › Añadir Punto de Interes

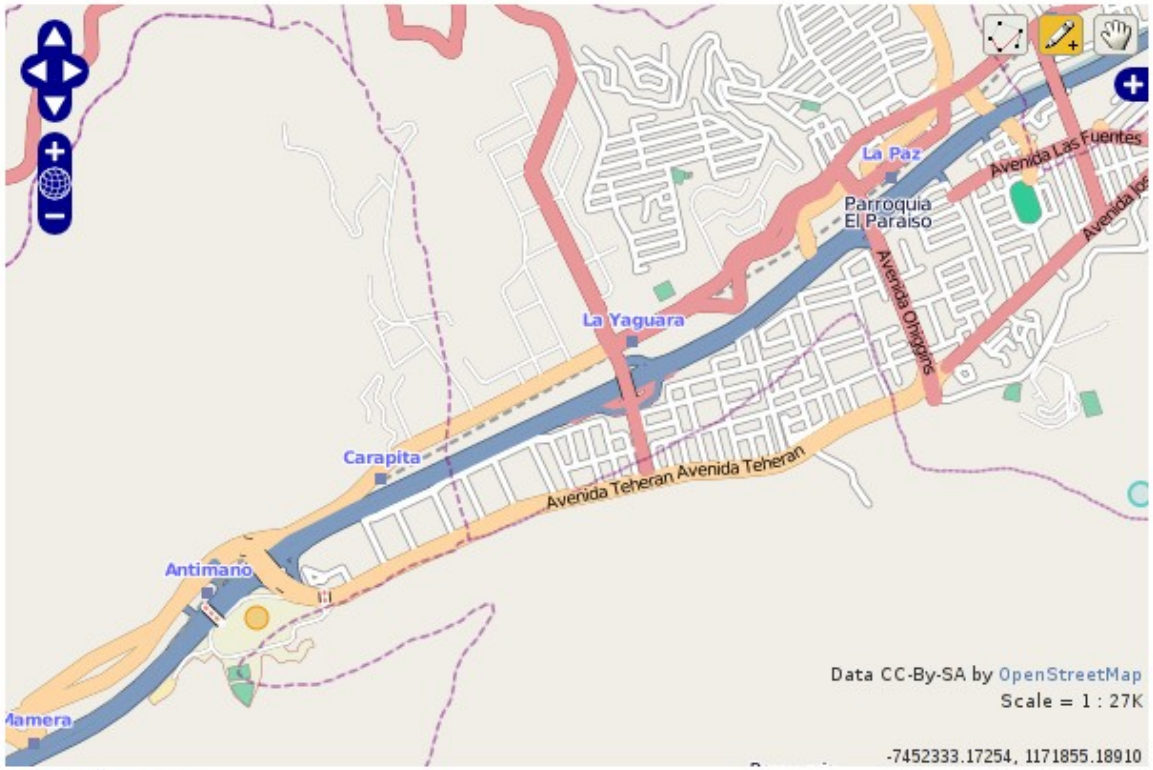
Añadir Punto de Interes

Nombre: PyCon Venezuela 2012

Descripcion:

Día 2. Sesión de Web Mapping

Pto:



Delete all Features

Grabar y añadir otro | Grabar y continuar editando | Grabar

Paso a Paso

- GeoAdmin

Paso a Paso

- Plantilla

```
(gdj) $ mkdir templates
(gdj) $ nano templates/PuntosKML.kml
....
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/{% block kml_version %}2.1{%
endblock %}">
<Document>
{% for punto in puntos %}<Placemark>
    <name>{{ punto.nombre }}</name>
    <description>{{ punto.descripcion }}</description>
    {{ punto.kml|safe }}
  </Placemark>{% endfor %}
</Document>
</kml>
....
(gdj) $
```

Paso a Paso

- Plantilla

```
(gdj) $ nano web_mapping/settings.py
....
TEMPLATE_DIRS = (
    "/home/(coloque la ruta de su pc)/web_mapping/templates",
)
....
(gdj) $
```

Paso a Paso

- Vista

```
(gdj) $ nano app1/views.py
....
from django.contrib.gis.shortcuts import render_to_kml
from models import PuntosDeInteres

def kml(request):
    puntos = PuntosDeInteres.objects.kml()
    return render_to_kml("PuntosKML.kml",{"puntos":puntos})
....
(gdj) $ nano web_mapping/urls.py
....
from django.conf.urls import patterns, include, url
from app1.views import kml
from django.contrib.gis import admin
admin.autodiscover()
urlpatterns = patterns('',
    url(r'^admin/', include(admin.site.urls)),
    url(r'^kml/$', kml),
)
....
```

Paso a Paso

- Vista

```
(gdj) $ python manage.py runserver
```

  - En su navegador visite http://localhost:8000/kml/
  - Vea el archivo generado

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
<Placemark>
    <name>PyCon Venezuela 2012</name>
    <description>Día 2. Sesión de Web Mapping</description>
    <Point><coordinates>-66.97651862,10.46459402</coordinates></Point>
  </Placemark>
</Document>
</kml>
```

Paso a Paso

- OpenLayers

OpenLayers

Home | Support & Development

Sponsorship | Documentation | Download | FAQ | Email Lists | Blog

OpenLayers: Free Maps for the Web

Get OpenLayers Now!

- 2.12 (Stable): .tar.gz | .zip
- 2.12 Release Notes
- API Documentation, User documentation
- See examples of OpenLayers Usage: Release Examples (2.12), Development Examples
- Fork us on GitHub

Put an open map widget in any web page!

About...

OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles and markers loaded from any source. OpenLayers has been developed to further the use of geographic information of all kinds. OpenLayers is completely free, Open Source JavaScript, released under the 2-clause BSD License (also known as the FreeBSD).

Supporting OpenLayers

OpenLayers is developed and supported by a number of organizations around the world. We are also looking for sponsors to help support the community. If you are in a position where you want to support the development of OpenLayers, but do not have development resources to share, you may be interested in supporting through our sponsorship program.

Books about OpenLayers

OpenLayers Cookbook

Antonio Santiago Perez

Available in Print and eBook formats from Packt Publishing.

OpenLayers 2.10 Beginner's Guide

Paso a Paso

- OpenLayers
  - KISS: Keep It Simple S...
  - La mejor manera de aprender OpenLayers es comparando los ejemplos con lo que queremos hacer.
  - Entre a la página http://www.openlayers.org/ y revise los ejemplos para OpenStreetMap (OSM) y KML
  - En la carpeta de las plantillas escriba un archivo llamado *inicio.html*

Paso a Paso

- OpenLayers

```
<html>
  <head> <title>Web Mapping con Django</title>
    <link rel="stylesheet"
      href="http://openlayers.org/dev/theme/default/style.cs"
      type="text/css">
    <script src="http://openlayers.org/dev/OpenLayers.js"></script>
    <script type="text/javascript">
          /* aquí va el script de OpenLayers */
    </script>
    <style>
      .map {border: 1px solid #ccc; width: 800px; height: 400px;  }
    </style>
  </head>
  <body onload="init()">
    <h1>Web Mapping con Django</h1>
    <hr />
    <div id="map" class="map"></div>
  </body>
</html>
```

Paso a Paso

- OpenLayers

```
var map, layer, kml;
function init(){
map = new OpenLayers.Map( 'map');
layer = new OpenLayers.Layer.OSM("OSM");
map.addLayer(layer);
kml = new OpenLayers.Layer.Vector("KML", {
  strategies: [new OpenLayers.Strategy.Fixed()],
  protocol: new OpenLayers.Protocol.HTTP({
  url: "kml/",
  format: new OpenLayers.Format.KML({
    extractStyles: true, extractAttributes: true, maxDepth: 2 })
  })  })
map.addLayer(kml);
map.setCenter(
  new OpenLayers.LonLat(-66.83, 10.45).transform(
  new OpenLayers.Projection("EPSG:4326"),
  map.getProjectionObject()), 8);
}
```

Paso a Paso

- Portal

```
(gdj) $ nano web_mapping/urls.py
....
from django.conf.urls import patterns, include, url
from django.views.generic.simple import direct_to_template
from app1.views import kml

from django.contrib.gis import admin
admin.autodiscover()

urlpatterns = patterns('',

    url(r'^admin/', include(admin.site.urls)),
    url(r'^kml/$', kml),
    url(r'^$', direct_to_template, {'template':'inicio.html'}),
)
....
(gdj) $
```

Paso a Paso

- Portal

Web Mapping con Django

Lo que sigue

- Agregar Controles al mapa
- Hacer un Popup que muestre los datos del punto de interés
- Agregar cambiar el símbolo del punto de interés
- Mejorar el diseño de la página
- Crear un método de captura sin el GeoAdmin
- Etc, etc, etc..

A modo de Conclusión

- Hacer Web Mapping con Django es directo, sin embargo hay muchos temas intermedios que deberá comenzar a manejar, como:
  - Bases de datos espaciales
  - Librería OpenLayers o Equivalente
  - Fuentes de Datos (servicios WMS, WFS, Google Maps, etc)
  - Proyecciones de mapas
  - Análisis Espacial

A modo de Conclusión

- Ya tiene unos primeros pasos adelantados, ahora aplique lo aquí visto en su caso de estudio.
- Además de los ejemplos de OpenLayers puede usar el proyecto Geoportal como una referencia o una plantilla para su proyecto particular
  - https://bitbucket.org/atmantree/geoportal/
- Prefiera fuentes y estándares libres.
- Disfrute mapeando la red.

Gracias por su tiempo..

Licencia del documento: CC BY-NC-SA 3.0
http://creativecommons.org/licenses/by-nc-sa/3.0/