

TUTORIAL

Rails GIS Hacks

Berlin | Germany

Shoaib Burq | Kashif Rasul

Monday, September 17, 2007

13:30 – 17:00



01

Using Geocoders

Geocoding

What is it?

- “the heart of most mapping applications”
Google Maps Apps. with Rails and Ajax by A. Lewis, et al.
- Convert any information to geographic identifiers for plotting on your map

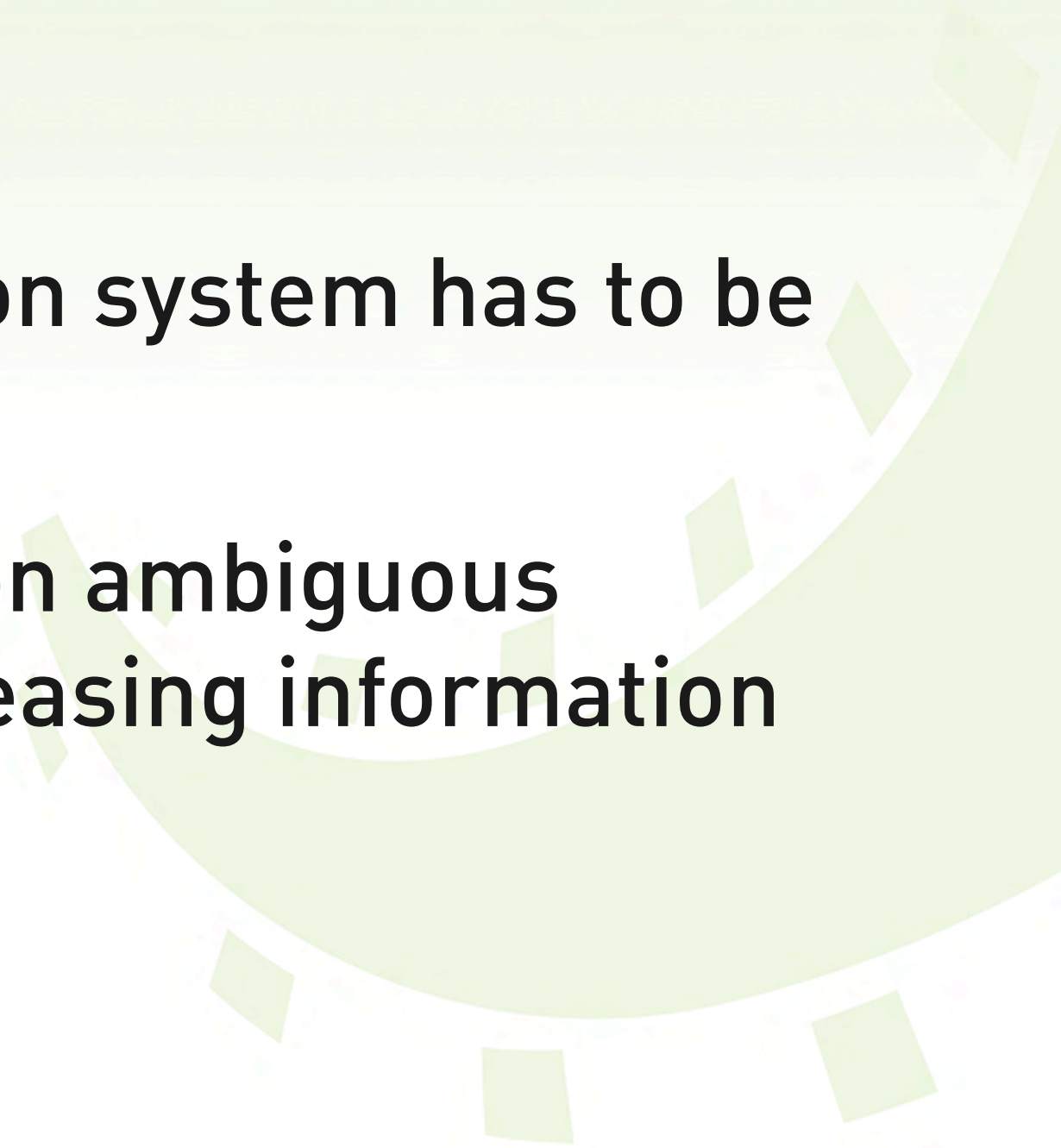
Geocoding

Example

- “Torstraße 104, 10119, Berlin, Germany”
→ lat = 52.530051, lng = 13.403495
- “85.178.26.159” → Berlin, Germany

Geocoding

Complications

- Geographic information system has to be updated regularly
 - Distinguishing between ambiguous addresses needs increasing information
- 

Geocoding Services

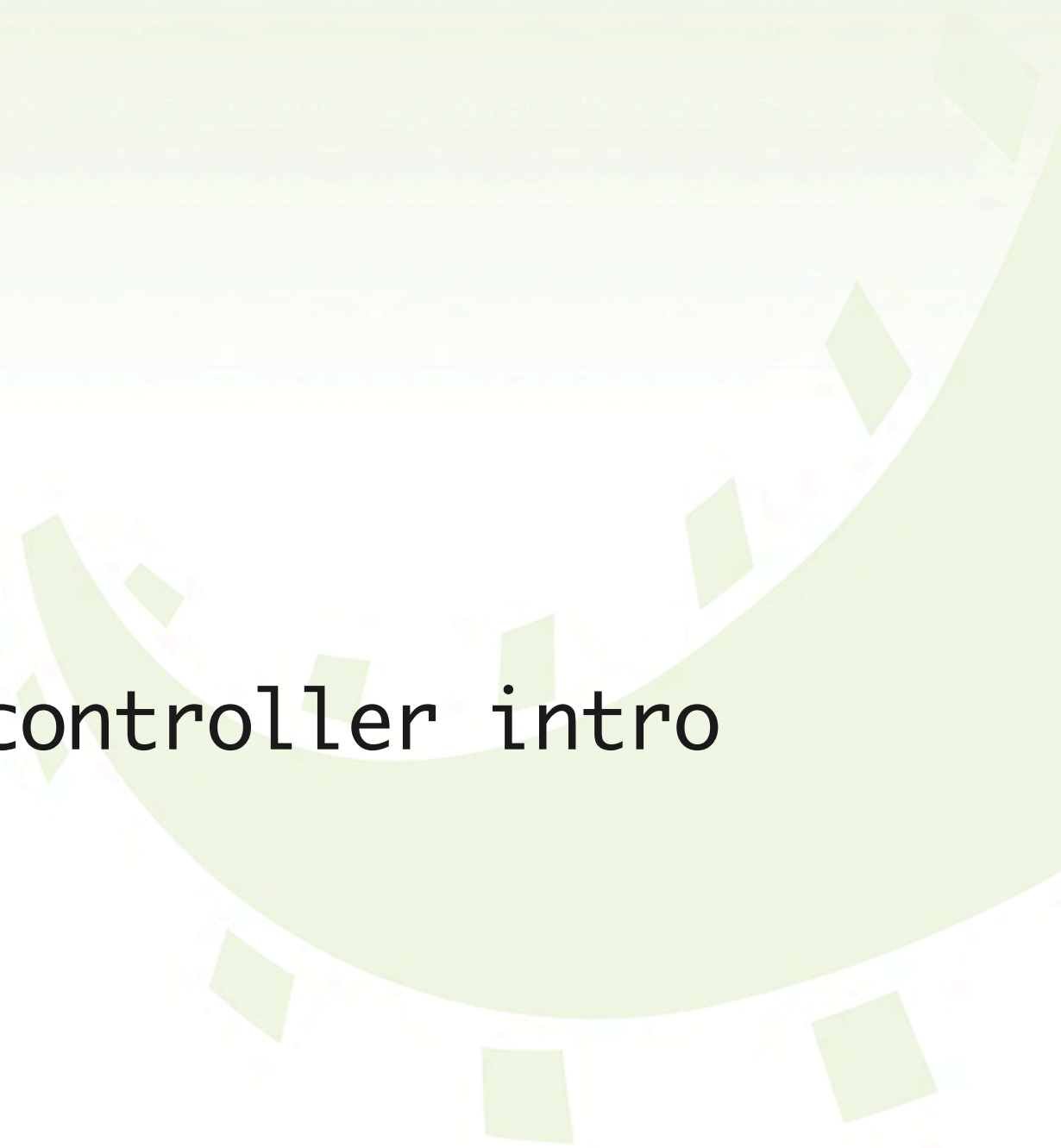
- Google Maps
 - Yahoo! Maps
 - Geocoder.us & Geocoder.ca
- 
- A decorative graphic in the bottom right corner consisting of a large, light green curved shape with several small, darker green squares scattered along its inner edge.

Intro to Yahoo! Maps with AJAX

```
$ rails yahoo
```

```
$ cd yahoo
```

```
$ ruby script/generate controller intro
```

A large, light green decorative shape resembling a gear or a stylized 'C' is positioned in the bottom right corner of the slide. It has several small, darker green rectangular tabs protruding from its outer edge.

app/views/intro/map.rhtml

```
<html>
  <head>
    <script type="text/javascript"
      src="http://api.maps.yahoo.com/ajaxymap?
v=3.7&appid=pPDCgjnV34FJ3TxysU9K.FpFYQ3A_QYJ4VrAJQuyFcFv91Hf0r3PU5tr3SYBhMvOoM_">
    </script>

    <script type="text/javascript">
      function load () {
        // Create a map object
        var map = new YMap(document.getElementById('map'));

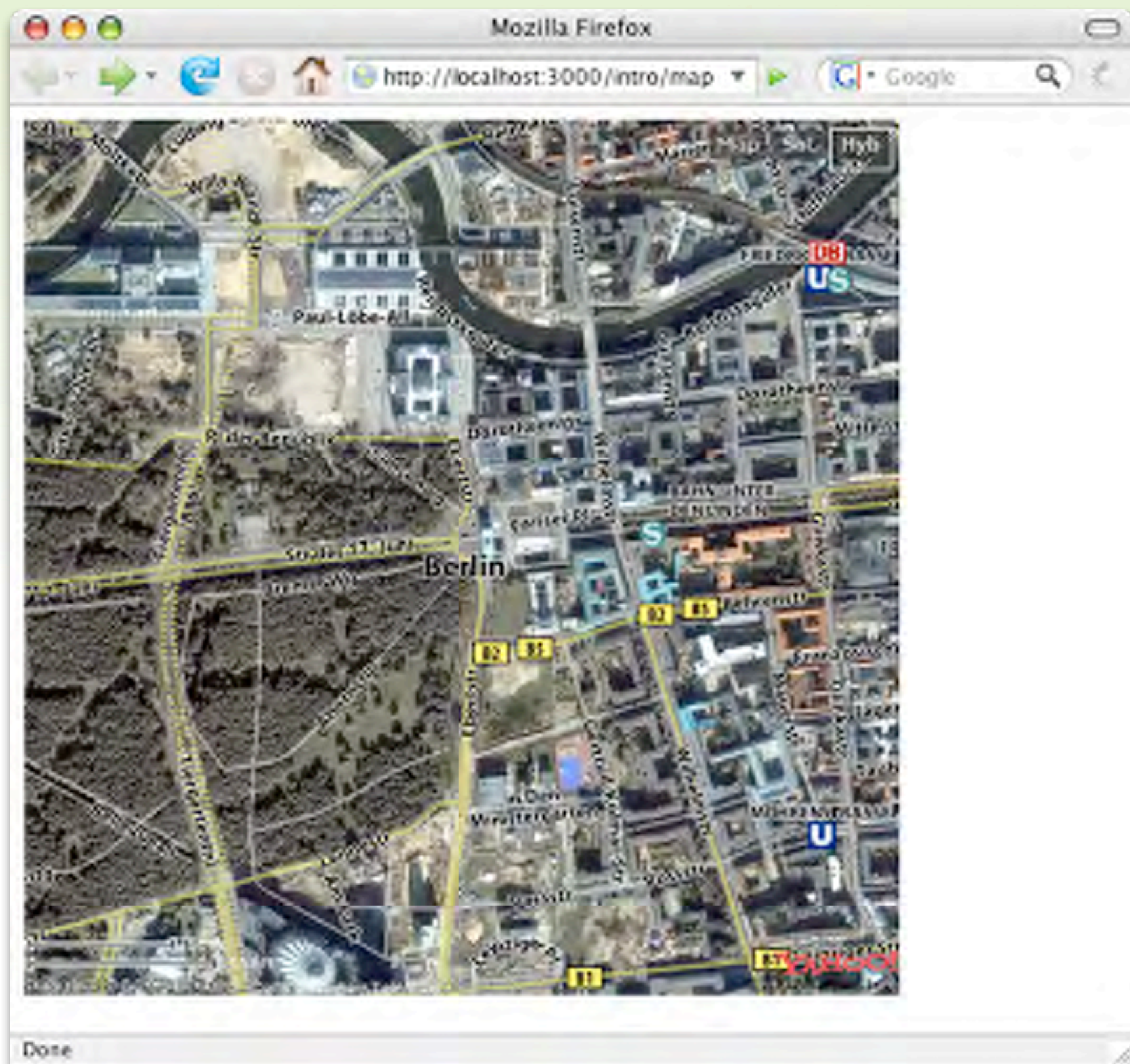
        // Add map type control
        map.addTypeControl();

        // Set map type to either of: YAHOO_MAP_SAT,
        // YAHOO_MAP_HYB, YAHOO_MAP_REG
        map.setMapType(YAHOO_MAP_HYB);

        // Display the map centered on a geocoded location
        map.drawZoomAndCenter("Berlin", 3);
      }
    </script>
  </head>
```

app/views/intro/map.rhtml

```
<body onload="load()">  
  <div id="map" style="width: 500px; height: 500px"></div>  
</body>  
</html>
```

Geocoding

Using REST web services

- Format a REST query
- Use OpenURI to send the query
- Parse the XML response to extract geo information using REXML

Google Maps' REST Geocoder

`http://maps.google.com/maps/geo?q=Torstra%C3%9Fe
+104,+Berlin,
+Germany&output=xml&key=ABQIAAAAwWqh7sPpuhNCdGZ0pieS
hBTJQa0g3IQ9GZqIMmInSLzwtGDKaBQ0JH6Tw4jIlz7bMDU6qtLF
_9TSHQ`

XML response

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Response>
    <name>Torstraße 104, Berlin, Germany</name>
    <Status>
      <code>200</code>
      <request>geocode</request>
    </Status>
    <Placemark id="p1">
      <address>Torstraße 104, 10119 Mitte, Berlin, Germany</address>
      <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0" Accuracy="8">
        <Country>
          <CountryNameCode>DE</CountryNameCode>
          <AdministrativeArea>
            <AdministrativeAreaName>Berlin</AdministrativeAreaName>
            <SubAdministrativeArea>
              <SubAdministrativeAreaName>Berlin</SubAdministrativeAreaName>
              <Locality>
                <LocalityName>Berlin</LocalityName>
                <DependentLocality>
                  <DependentLocalityName>Mitte</DependentLocalityName>
                  <Thoroughfare>
                    <ThoroughfareName>Torstraße 104</ThoroughfareName>
                  </Thoroughfare>
                  <PostalCode>
                    <PostalCodeNumber>10119</PostalCodeNumber>
                  </PostalCode>
                </DependentLocality>
              </Locality>
            </SubAdministrativeArea>
          </AdministrativeArea>
        </Country>
      </AddressDetails>
      <Point>
        <coordinates>13.403424,52.529717,0</coordinates>
      </Point>
    </Placemark>
  </Response>
</kml>
```


Rails plugins

An initial comparison

- GeoKit
 - Graticule & acts_as_geocodable
 - ActsAsLocatable
 - YM4R
- 

GeoKit install

```
$ rails geokit
```

```
$ cd geokit
```

```
$ ruby script/plugin install  
svn://rubyforge.org/var/svn/geokit/trunk
```

API keys in config/envirnement.rb

This is your yahoo application key for the Yahoo Geocoder.

See <http://developer.yahoo.com/faq/index.html#appid>

and <http://developer.yahoo.com/maps/rest/V1/geocode.html>

GeoKit::Geocoders::yahoo =

'pPDCgjnV34FJ3TxysU9K.FpFYQ3A_QYJ4VrAJQuyFcFv91Hf0r3PU5tr3SYBhMv0oM__'

This is your Google Maps geocoder key.

See <http://www.google.com/apis/maps/signup.html>

and http://www.google.com/apis/maps/documentation/#Geocoding_Examples

GeoKit::Geocoders::google =

'ABQIAAAAwWqh7sPpuhNCdGZ0pieShBTJQa0g3IQ9GZqIMmInSLzwtGDKaBQ0JH6Tw4jIlz
7bMDU6qtLF_9TSHQ'

Geocoding from multiple providers

```
GeoKit::Geocoders::provider_order = [:yahoo, :google]
```

Test it out

```
$ ruby script/console  
Loading development environment.
```

```
>> include GeoKit::Geocoders  
=> Object
```

```
>> home = MultiGeocoder.geocode("Torstrasse 104, 10119, Berlin,  
Germany")  
=> #<GeoKit::GeoLoc:0x35de820 @lat=52.530051, @state="Germany",  
@street_address="Torstrasse 104", @country_code="DE",  
@provider="yahoo", @precision="address", @zip=nil,  
@lng=13.403495, @city="10119 Mitte", @success=true>
```

```
>> home.lat  
=> 52.530051
```

```
>> home.lng  
=> 13.403495
```

In memory calculations

```
>> office = MultiGeocoder.geocode("Lepsiusstrasse 70, Steglitz, Berlin,
Germany")
=> #<GeoKit::GeoLoc:0x341e5f8 @lat=52.460126, @state="Germany",
@street_address="Lepsiusstrasse 70", @country_code="DE", @provider="yahoo",
@precision="address", @zip=nil, @lng=13.316571, @city="12163 Steglitz",
@success=true>

>> office.distance_to(home, :units => :kms)
=> 9.75995820357575

>> heading = home.heading_to(office) # result is in degrees, 0 is north
=> 217.15430202928

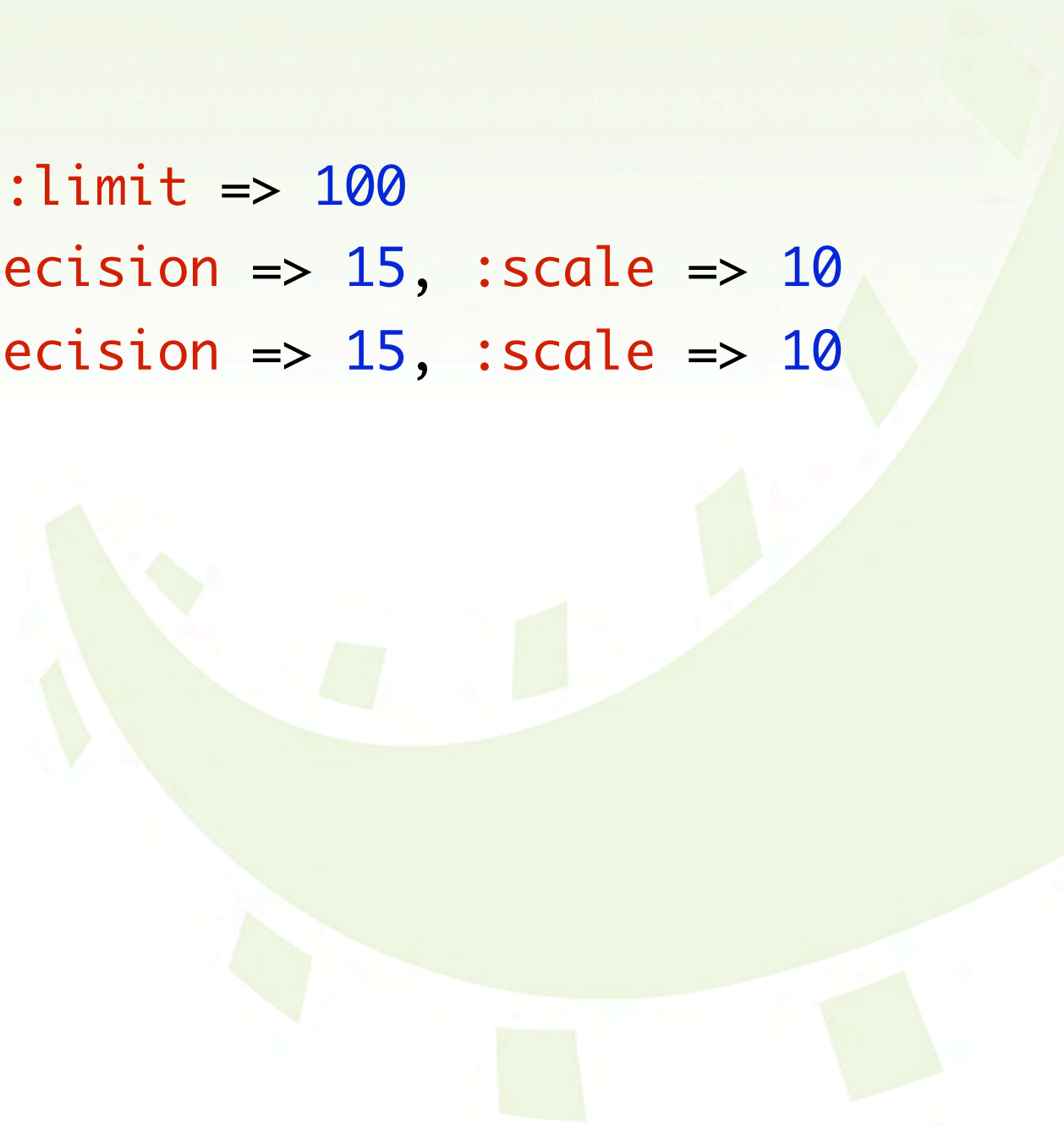
>> endpoint = home.endpoint(90, 2) # given a heading (east) and distance
=> #<GeoKit::LatLng:0x33f6878 @lat=52.5300414818178, @lng=13.4510238774836>

>> midpoint = home.midpoint_to(office)
=> #<GeoKit::LatLng:0x33f08b0 @lat=52.4950964615994, @lng=13.3599984433113>
```


Auto geocoding a model's address on create

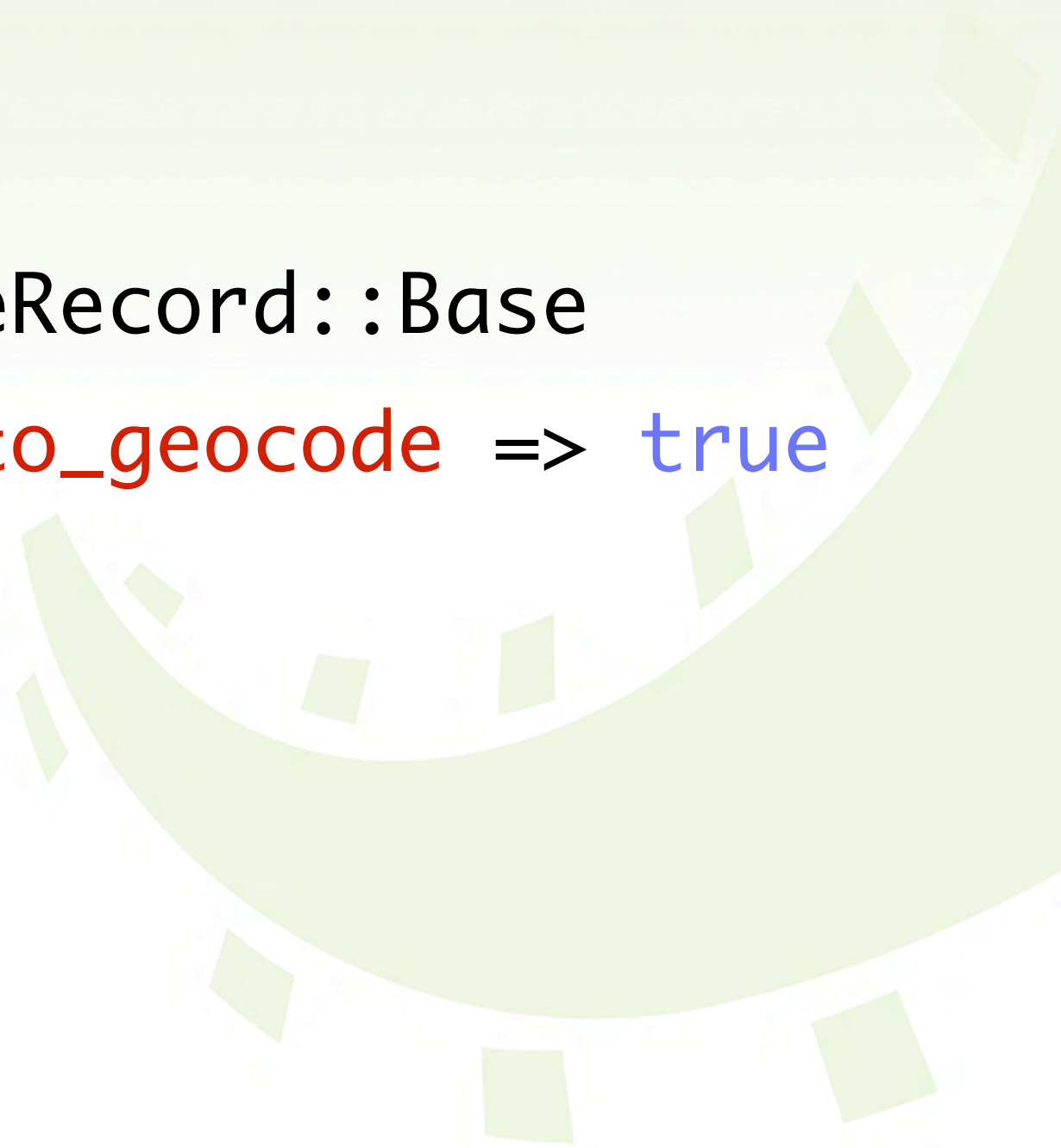
```
class CreateLocations < ActiveRecord::Migration
  def self.up
    create_table :locations do |t|
      t.column :address, :string, :limit => 100
      t.column :lat, :decimal, :precision => 15, :scale => 10
      t.column :lng, :decimal, :precision => 15, :scale => 10
    end
  end

  def self.down
    drop_table :locations
  end
end
```



app/models/location.rb

```
class Location < ActiveRecord::Base  
  acts_as_mappable :auto_geocode => true  
end
```



Testing this out

```
>> Location.find :all  
=> []
```

```
>> Location.create(:address => "Torstrasse 104, Berlin, Germany")  
=> #<Location:0x344d074 @errors=#<ActiveRecord::Errors:0x341c99c @errors=  
{}, @base=#<Location:0x344d074 ...>, @attributes={"id"=>4,  
"lng"=>13.403495, "lat"=>52.530051, "address"=>"Torstrasse 104, Berlin,  
Germany"}, @new_record=false>
```

```
>> home = Location.find :first  
=> #<Location:0x3416e34 @attributes={"lng"=>#<BigDecimal:  
3416e5c,'0.13403495E2',12(16)>, "id"=>"4", "lat"=>#<BigDecimal:  
3416e84,'0.52530051E2',12(16)>, "address"=>"Torstrasse 104, Berlin,  
Germany"}>
```

```
>> Location.create(:address => "Lepsiusstrasse 70, Berlin, Germany")  
=> #<Location:0x3413608 @errors=#<ActiveRecord::Errors:0x33e52f8 @errors=  
{}, @base=#<Location:0x3413608 ...>, @attributes={"id"=>5,  
"lng"=>13.316571, "lat"=>52.460126, "address"=>"Lepsiusstrasse 70, Berlin,  
Germany"}, @new_record=false>
```


Sort by distance

```
>> locs = Location.find :all
```

```
=> [  
  #<Location:0x3375d90  
    @attributes={"lng"=>#<BigDecimal:3375f20,'0.13403495E2',12(16)>, "id"=>"4",  
"lat"=>#<BigDecimal:3375f48,'0.52530051E2',12(16)>, "address"=>"Torstrasse 104,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d68  
    @attributes={"lng"=>#<BigDecimal:3375e94,'0.13316571E2',12(16)>, "id"=>"5",  
"lat"=>#<BigDecimal:3375ea8,'0.52460126E2',12(16)>, "address"=>"Lepsiusstrasse 70,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d40  
    @attributes={"lng"=>#<BigDecimal:3375e1c,'0.13365749E2',12(16)>, "id"=>"6",  
"lat"=>#<BigDecimal:3375e30,'0.5249112E2',12(16)>, "address"=>"Crellestrasse 23,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d18  
    @attributes={"lng"=>#<BigDecimal:3375da4,'0.13386817E2',12(16)>, "id"=>"7",  
"lat"=>#<BigDecimal:3375db8,'0.52510553E2',12(16)>, "address"=>"Mauerstrasse 65,  
Berlin, Germany"}>  
]
```

sort_by_distance_from

```
>> locs.sort_by_distance_from(home)
```

```
=> [  
  #<Location:0x3375d90 @distance=0.0,  
    @attributes={"lng"=>#<BigDecimal:3375f20,'0.13403495E2',12(16)>, "id"=>"4",  
"lat"=>#<BigDecimal:3375f48,'0.52530051E2',12(16)>, "address"=>"Torstrasse 104,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d18 @distance=1.52043248966975,  
    @attributes={"lng"=>#<BigDecimal:3375da4,'0.13386817E2',12(16)>, "id"=>"7",  
"lat"=>#<BigDecimal:3375db8,'0.52510553E2',12(16)>, "address"=>"Mauerstrasse 65,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d40 @distance=3.12676959370349,  
    @attributes={"lng"=>#<BigDecimal:3375e1c,'0.13365749E2',12(16)>, "id"=>"6",  
"lat"=>#<BigDecimal:3375e30,'0.5249112E2',12(16)>, "address"=>"Crellestrasse 23,  
Berlin, Germany"}>,  
  
  #<Location:0x3375d68 @distance=6.06585345156976,  
    @attributes={"lng"=>#<BigDecimal:3375e94,'0.13316571E2',12(16)>, "id"=>"5",  
"lat"=>#<BigDecimal:3375ea8,'0.52460126E2',12(16)>, "address"=>"Lepsiusstrasse 70,  
Berlin, Germany"}>  
]
```

AR drops the distance column

```
>> locs = Location.find :all, :origin=>home, :within=>5, :order=>'distance'

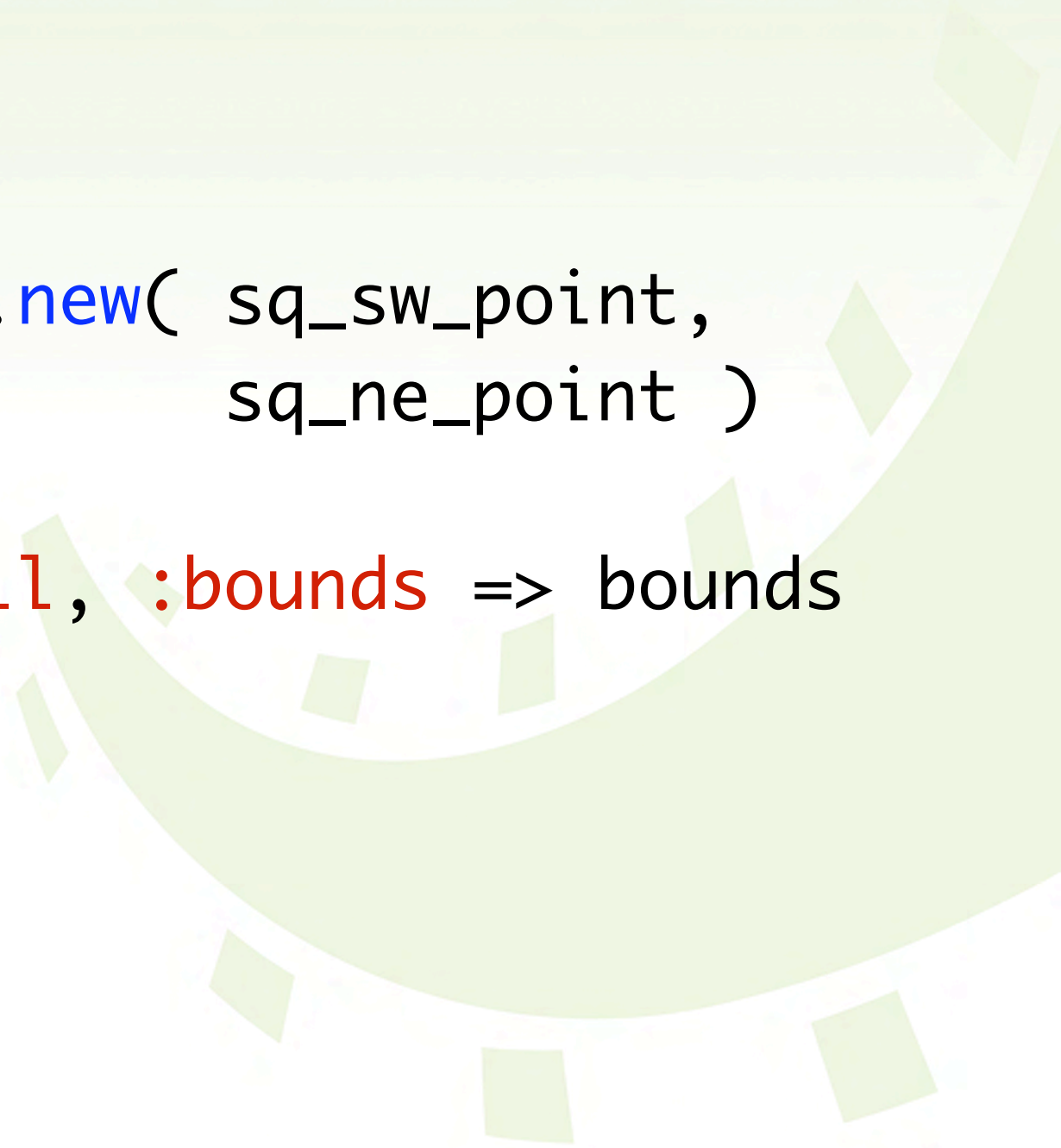
=> [
  #<Location:0x3362268 @attributes={"lng"=>#<BigDecimal:
3362394,'0.13403495E2',12(16)>, "id"=>"4", "lat"=>#<BigDecimal:
33623bc,'0.52530051E2',12(16)>, "address"=>"Torstrasse 104, Berlin, Germany",
"distance"=>"0"}>,

  #<Location:0x3362240 @attributes={"lng"=>#<BigDecimal:
33622f4,'0.13386817E2',12(16)>, "id"=>"7", "lat"=>#<BigDecimal:
3362308,'0.52510553E2',12(16)>, "address"=>"Mauerstrasse 65, Berlin, Germany",
"distance"=>"1.52043248966975"}>,

  #<Location:0x3362218 @attributes={"lng"=>#<BigDecimal:
336227c,'0.13365749E2',12(16)>, "id"=>"6", "lat"=>#<BigDecimal:
3362290,'0.5249112E2',12(16)>, "address"=>"Crellestrasse 23, Berlin, Germany",
"distance"=>"3.1267695948189"}>
]
```


GeoKit::Bounds

```
>> bounds = GeoKit::Bounds.new( sq_sw_point,  
                                   sq_ne_point )  
  
>> locs = Location.find :all, :bounds => bounds
```



Eager Loading

```
>> locs = Location.find :all,  
                        :origin => home,  
                        :include => [:reviews],  
                        :within => 5,  
                        :order => 'distance'
```

IP address geocoding

```
>> location = GeoKit::Geocoders::IpGeocoder.geocode('85.178.26.159')
```

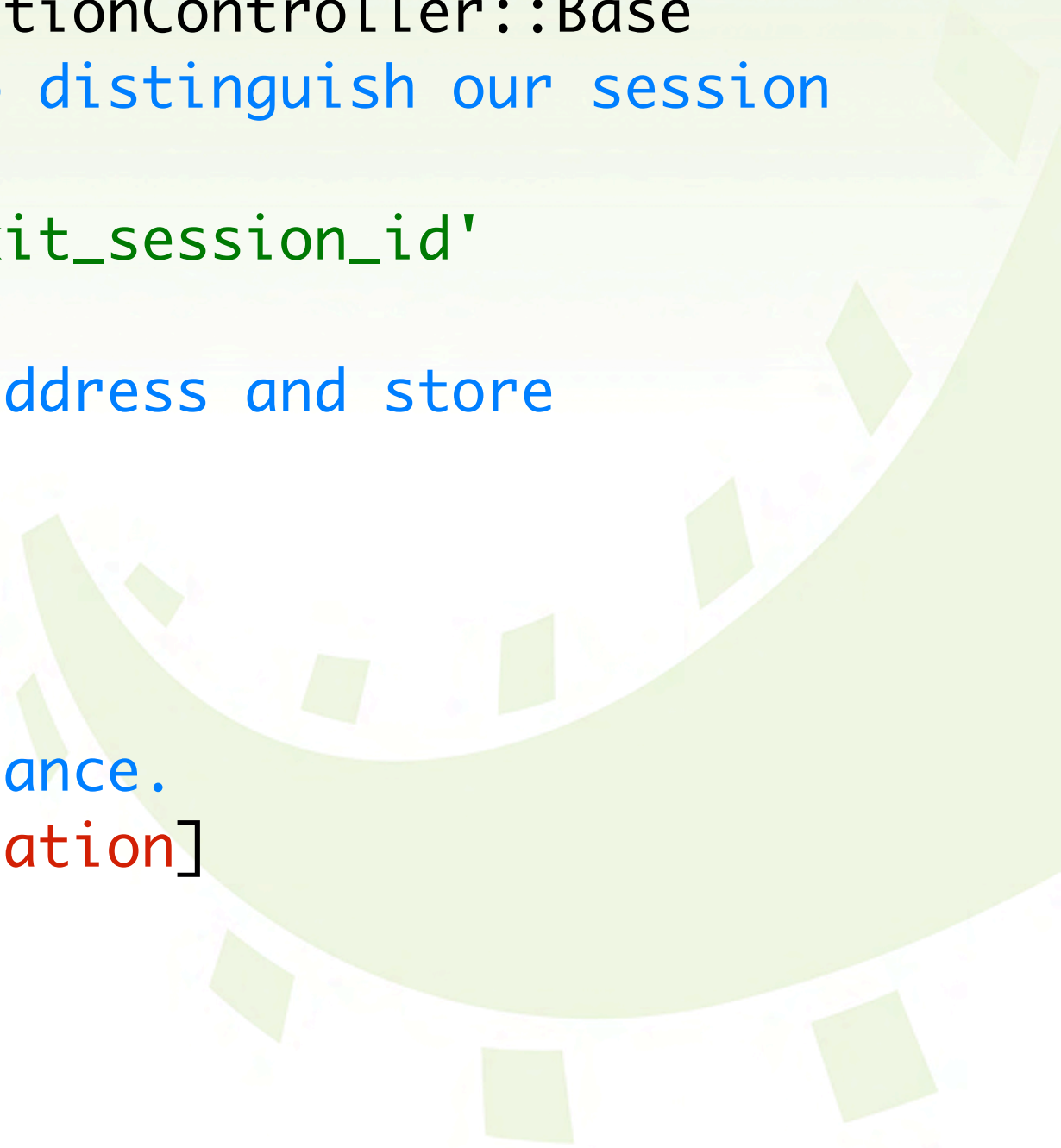
```
=> #<GeoKit::GeoLoc:0x3756fe0 @lat=52.5, @state=nil,  
@street_address=nil, @country_code="DE", @provider="hostip",  
@precision="unknown", @zip=nil, @lng=13.4167, @city="Berlin",  
@success=true>
```


Cached IP location in app/controllers/application.rb

```
class ApplicationController < ActionController::Base
  # Pick a unique cookie name to distinguish our session
  # data from others'
  session :session_key => '_geokit_session_id'

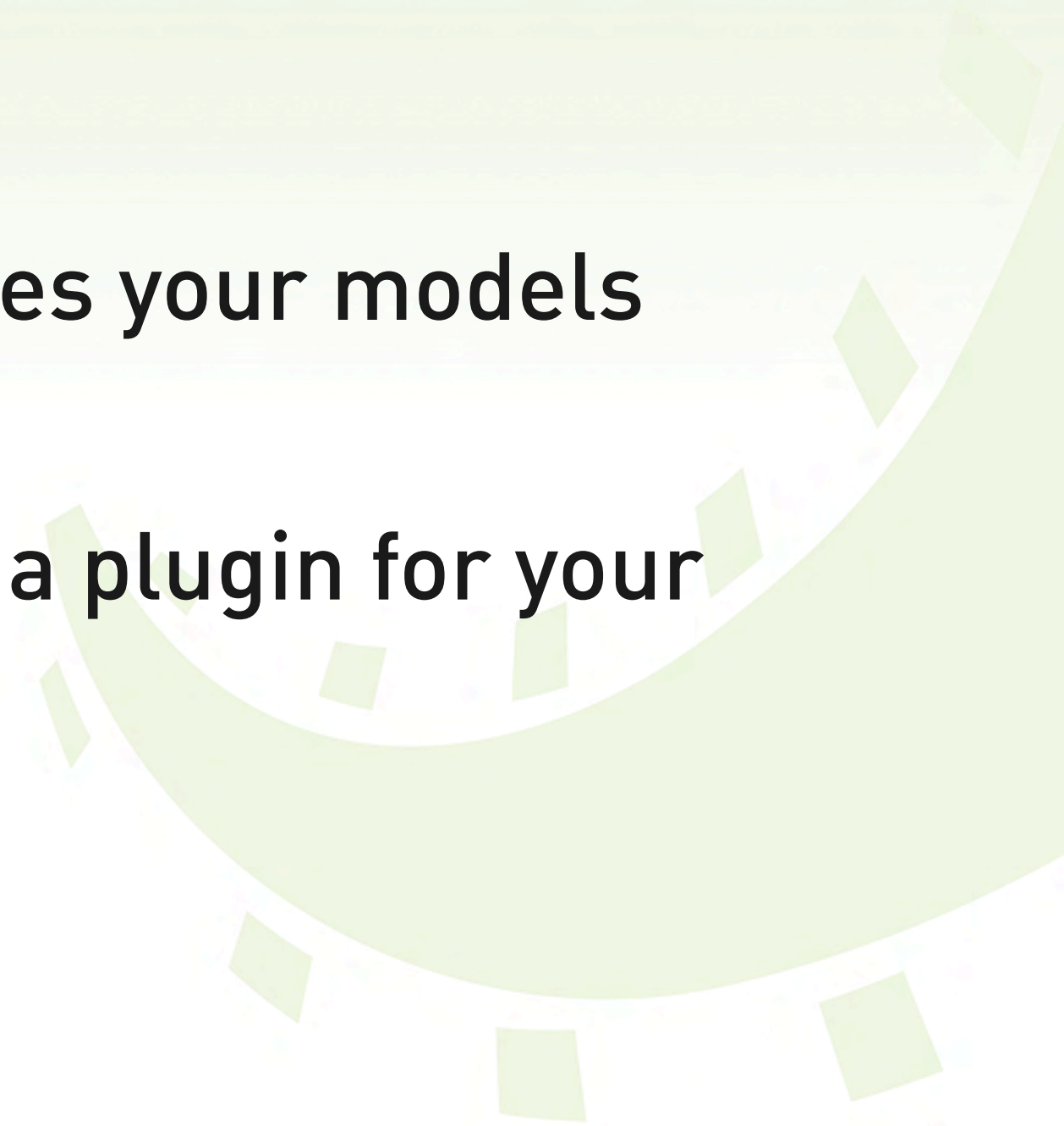
  # Auto-geocode the user's ip address and store
  # it in the session.
  geocode_ip_address

  def loc
    # @location is a GeoLoc instance.
    @location = session[:geo_location]
  end
end
```



Graticule

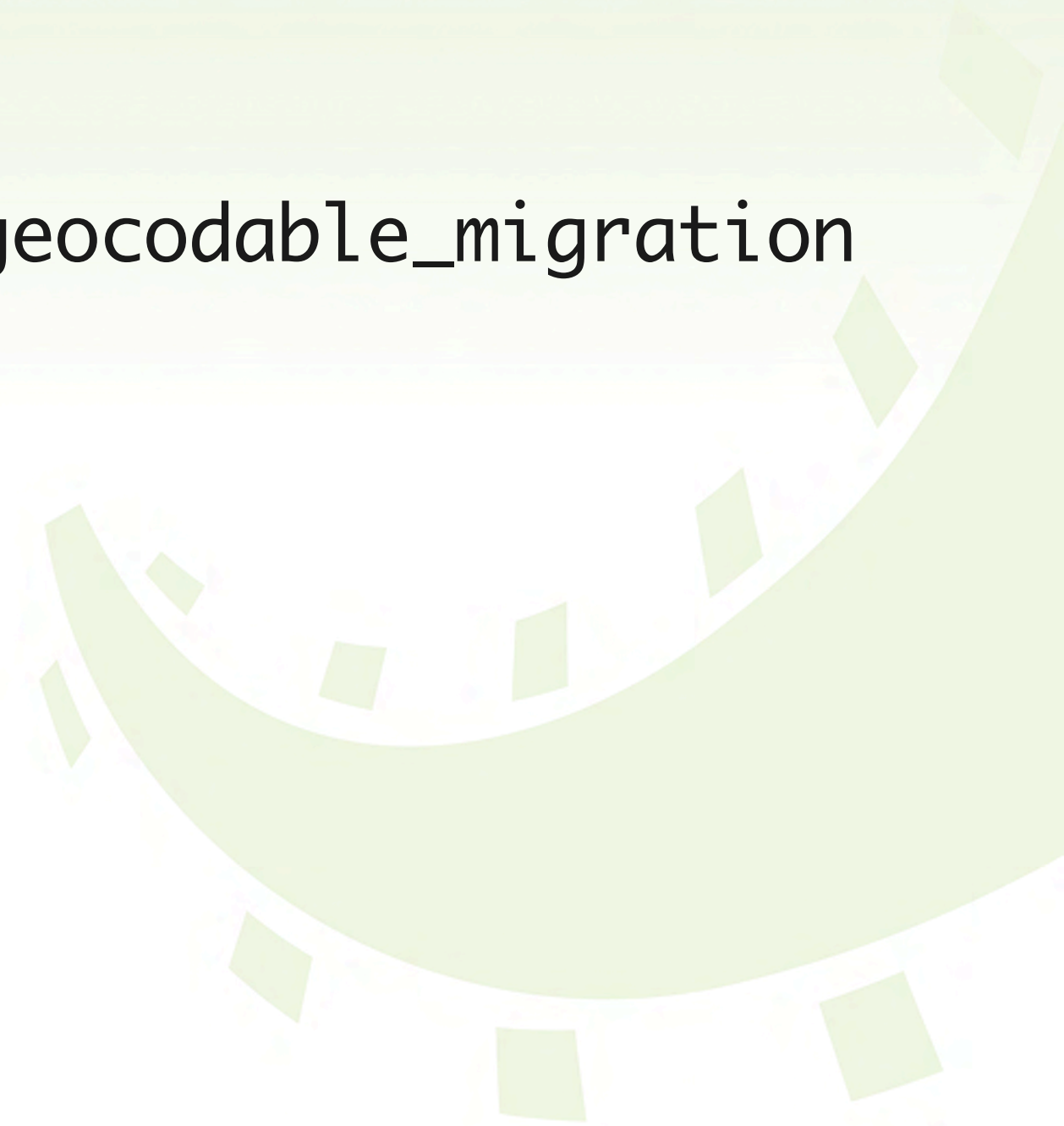
and its Rails plugin

- Automatically geocodes your models when they are saved
 - Comes as a gem plus a plugin for your Rails app
- 

Initial config

```
$ ruby script/generate geocodable_migration  
add_geocodable_tables
```

```
$ rake db:migrate
```

A large, stylized green gear graphic is positioned in the bottom right corner of the slide. The gear is partially visible, showing its teeth and a curved body. It has a light green color with a slight gradient and a soft shadow.

Multi geocoder & key in config/envirnement.rb

```
Geocode.geocoder = Graticule.service(:multi).new (

  Graticule.service(:yahoo).new (
    'pPDCgjnV34FJ3TxysU9K.FpFYQ3A_QYJ4VrAJQuyFcFv91Hf0r3
    PU5tr3SYBhMv0oM__'
  ),

  Graticule.service(:google).new (
    'ABQIAAAAwWqh7sPpuhNCdGZ0pieShBTJQa0g3IQ9GZqIMmInSLz
    wtGDKaBQ0JH6Tw4jIlz7bMDU6qtLF_9TSHQ'
  )
)
```

Model with required attributes

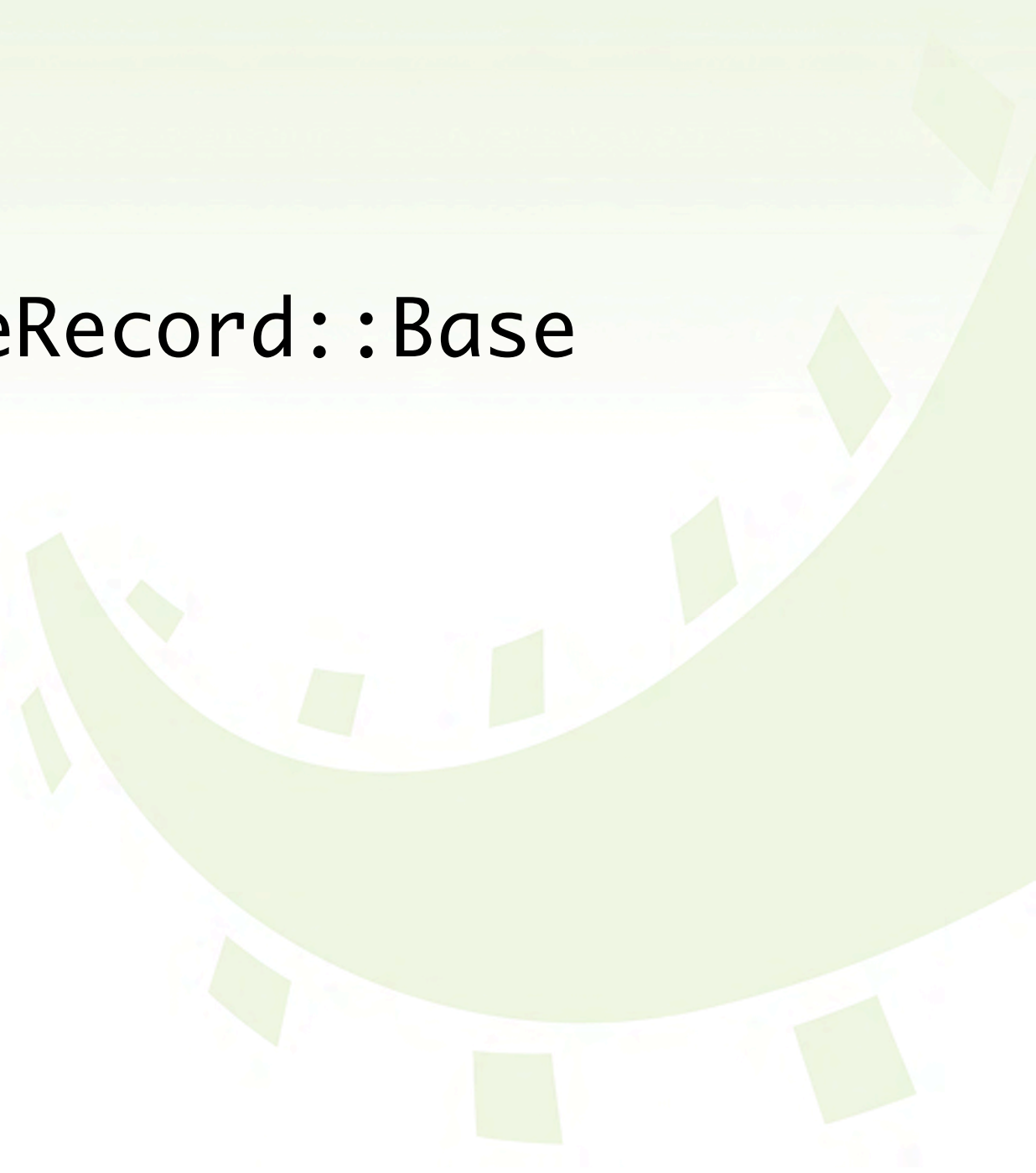
```
class CreateLocations < ActiveRecord::Migration
  def self.up
    create_table :locations do |t|
      t.column "street",      :string
      t.column "locality",    :string
      t.column "region",      :string
      t.column "postal_code", :string
      t.column "country",     :string
    end
  end

  def self.down
    drop_table :locations
  end
end
```



app/model/location.rb

```
class Location < ActiveRecord::Base  
  acts_as_geocodable  
end
```



Again we test it out

```
$ ruby script/console  
Loading development environment.
```

```
>> Location.find :all  
=> []
```

```
>> conf = Location.create :street => "Friedrichstrasse 151", :locality => "Berlin"  
=> #<Location:0x357ec40 @geocoding=#<Geocoding:0x356e9a8  
@errors=#<ActiveRecord::Errors:0x356dd78 @errors={}, @base=#<Geocoding:  
0x356e9a8 ...>>, @geocode=#<Geocode:0x357490c @attributes={"postal_code"=>nil,  
"latitude"=>#<BigDecimal:35749d4,'0.5251818E2',12(20)>, "region"=>"Germany",  
"country"=>"DE", "id"=>"2", "locality"=>"10117 Mitte", "street"=>"Friedrichstrasse  
151", "query"=>"Friedrichstrasse 151\nBerlin, ", "longitude"=>#<BigDecimal:  
35749ac,'0.13388423E2',12(20)>}>, @attributes={"geocodable_type"=>"Location", "id"=>4,  
"geocodable_id"=>4, "geocode_id"=>2}, @new_record=false>,  
@errors=#<ActiveRecord::Errors:0x357ccd8 @errors={}, @base=#<Location:0x357ec40 ...>>,  
@attributes={"postal_code"=>nil, "region"=>"Germany", "country"=>"DE", "id"=>4,  
"locality"=>"Berlin", "street"=>"Friedrichstrasse 151"}, @new_record=false>
```

```
>> conf.geocode.latitude  
=> #<BigDecimal:35749d4,'0.5251818E2',12(20)>
```

```
>> conf.geocode.longitude  
=> #<BigDecimal:35749ac,'0.13388423E2',12(20)>
```

distance_to

```
>> prevConf = Location.create :street => "777 NE Martin Luther King, Jr.  
Blvd.", :locality => "Portland", :region => "Oregon", :postal_code => 97232
```

```
=> #<Location:0x355c924 @geocoding=#<Geocoding:0x3555e6c  
@errors=#<ActiveRecord::Errors:0x355578c @errors={}, @base=#<Geocoding:  
0x3555e6c ...>>, @geocode=#<Geocode:0x3557cd0 @attributes=  
{"postal_code"=>"97232-2742", "latitude"=>#<BigDecimal:  
3557d98, '0.45528468E2', 12(20)>, "region"=>"OR", "country"=>"US", "id"=>"1",  
"locality"=>"Portland", "street"=>"777 Ne M L King Blvd", "query"=>"777 NE  
Martin Luther King, Jr. Blvd.\nPortland, Oregon 97232",  
"longitude"=>#<BigDecimal:3557d70, '-0.122661895E3', 12(20)>}>, @attributes=  
{"geocodable_type"=>"Location", "id"=>5, "geocodable_id"=>5,  
"geocode_id"=>1}, @new_record=false>, @errors=#<ActiveRecord::Errors:  
0x355b894 @errors={}, @base=#<Location:0x355c924 ...>>, @attributes=  
{"postal_code"=>97232, "region"=>"Oregon", "country"=>"US", "id"=>5,  
"locality"=>"Portland", "street"=>"777 NE Martin Luther King, Jr. Blvd."},  
@new_record=false>
```

```
>> conf.distance_to prevConf  
=> 5185.541406646
```

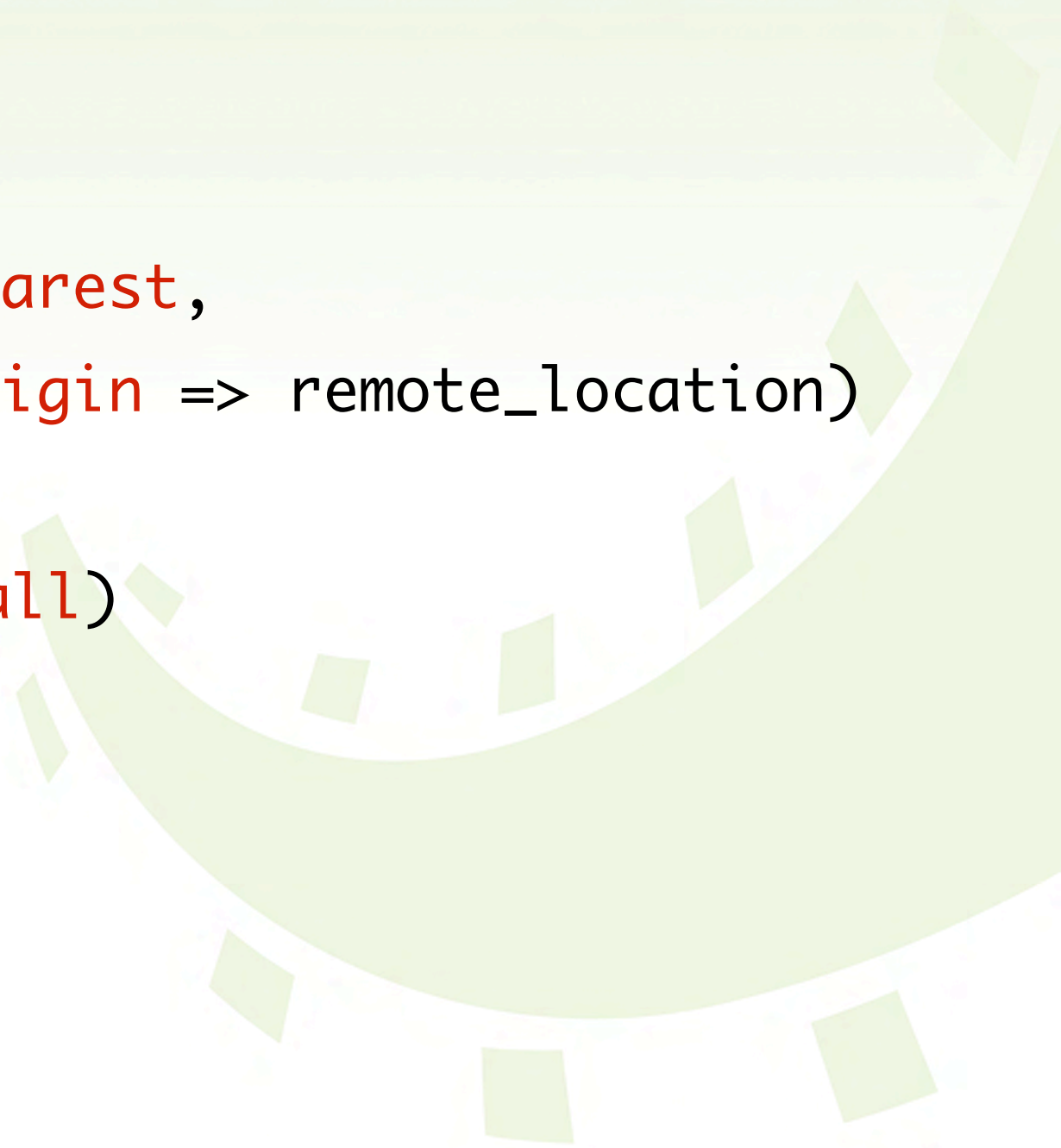
search by location

```
>> Location.find(:all, :within => 50, :origin =>
  "Torstrasse 104, Berlin, Germany")
```

```
=> [#<Location:0x35239f8 @readonly=true, @attributes=
  {"postal_code"=>nil, "region"=>"Germany",
  "country"=>"DE", "id"=>"4", "locality"=>"Berlin",
  "street"=>"Friedrichstrasse 151",
  "distance"=>"1.03758608910963"}>]
```

IP geocoding

```
def index
  @nearest = Location.find( :nearest,
                           :origin => remote_location)
  if remote_location
    @locations = Location.find(:all)
  end
end
```



02

Location data in ActiveRecord (PostGIS/PostgreSQL)

GeoRails Stack

Overview

YM4R

Spatial Adapter

GeoRuby

PostGIS Database

Prerequisites

Installing PostGIS

- Install PostgreSQL without PostGIS extension
 - Install PostGIS separately
 - Ensures you get the latest PostGIS release
- 
- A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has a thick orange outer ring and a lighter orange inner section. The gear's teeth are represented by small, rectangular orange blocks protruding from the outer ring.

Prerequisites

template_postgis

- create a clone of template1
 - add pl/pgsql language
 - install the postgis functions/libraries
 - grant role-based permissions
 - vacuum freeze
- 

template_postgis code

```
\c template1
```

```
CREATE DATABASE template_postgis with template = template1;
```

```
-- set the 'datistemplate' record in the 'pg_database' table for  
-- 'template_postgis' to TRUE indicating its a template
```

```
UPDATE pg_database SET datistemplate = TRUE where datname = 'template_postgis';
```

```
\c template_postgis
```

```
CREATE LANGUAGE plpgsql;
```

```
\i /usr/share/postgresql/contrib/lwpostgis.sql
```

```
\i /usr/share/postgresql/contrib/spatial_ref_sys.sql
```

```
-- set role based permissions in production env.
```

```
GRANT ALL ON geometry_columns TO PUBLIC;
```

```
GRANT ALL ON spatial_ref_sys TO PUBLIC;
```

```
-- vacuum freeze: it will guarantee that all rows in the database are  
-- "frozen" and will not be subject to transaction ID wraparound  
-- problems.
```

```
VACUUM FREEZE;
```

Prerequisites

GeoRuby

- Data structs to hold Spatial Data from PostGIS/MySQL-Spatial
- Based on OGC Simple Features Spec
- No geometry operations or reprojections
- Allows geo-data I/O into a number of useful formats

GeoRuby



```
$ sudo gem install georuby -y
```

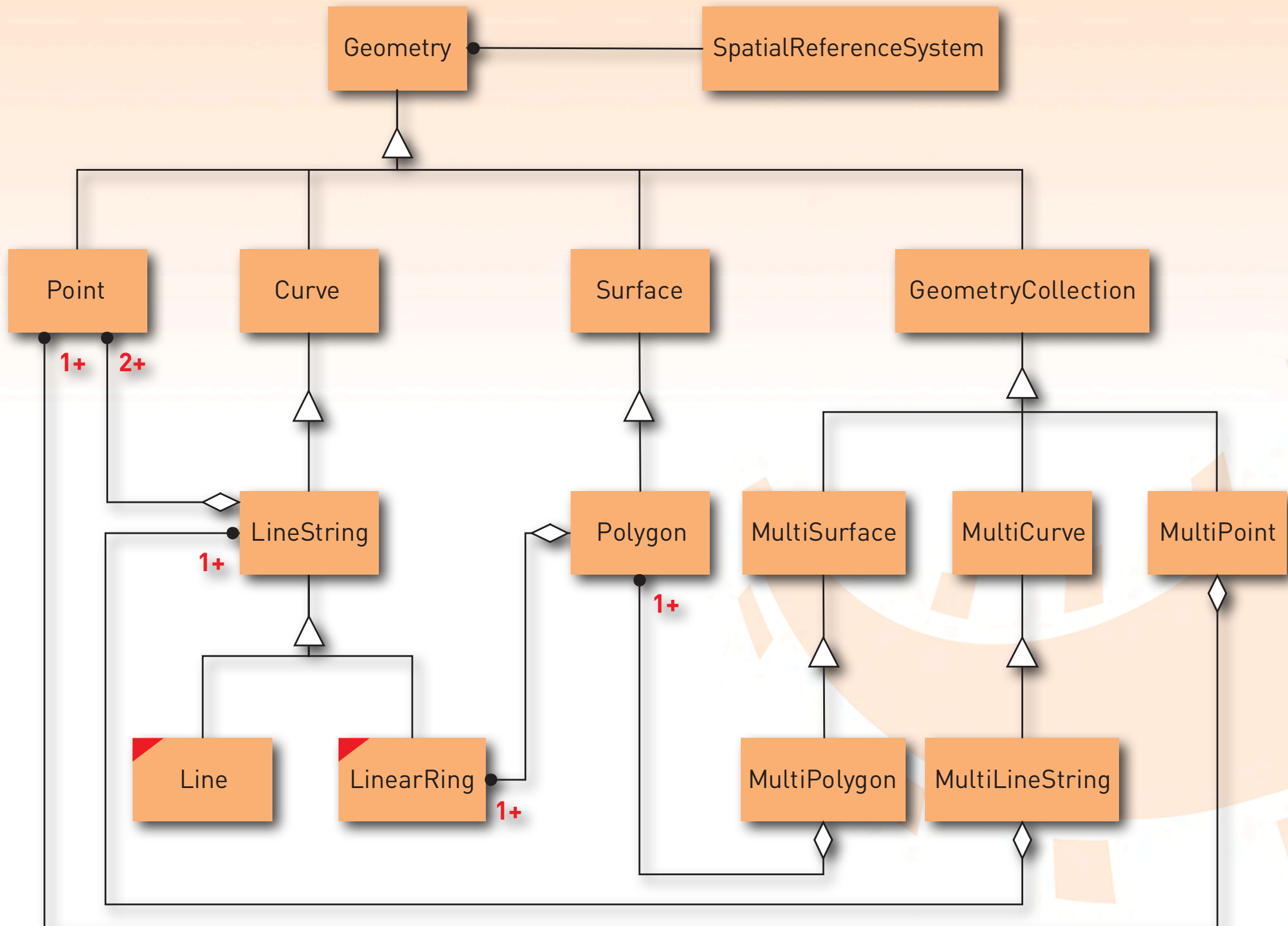
Geospatial Domain

What's so special about spatial?

- Point, Line, Polygon, Multi*



OGC simple features



Vector data

<http://www.ngdc.noaa.gov/mgg/shorelines>



National Geophysical Data Center (NGDC)
NOAA Satellite and Information Service

[NOAA > NESDIS > NGDC > MGG > bathymetry](#)[privacy policy](#)

[All MGG](#) [Coastal](#) [Fishing](#) [Global](#) [Lakes](#) [Multibeam](#) [NOS surveys](#) [Trackline](#)

Marine Geology & Geophysics

Shoreline / Coastline Data



 [Coastline Extractor](#)

Geospatial Domain

What's so special about spatial?

- Point, Line, Polygon, Multi*
- Raster Data



Raster data

<http://edc.usgs.gov/products/elevation/gtopo30/gtopo30.html>



science for a changing world

Earth Resources Observation and Science (EROS)

Products

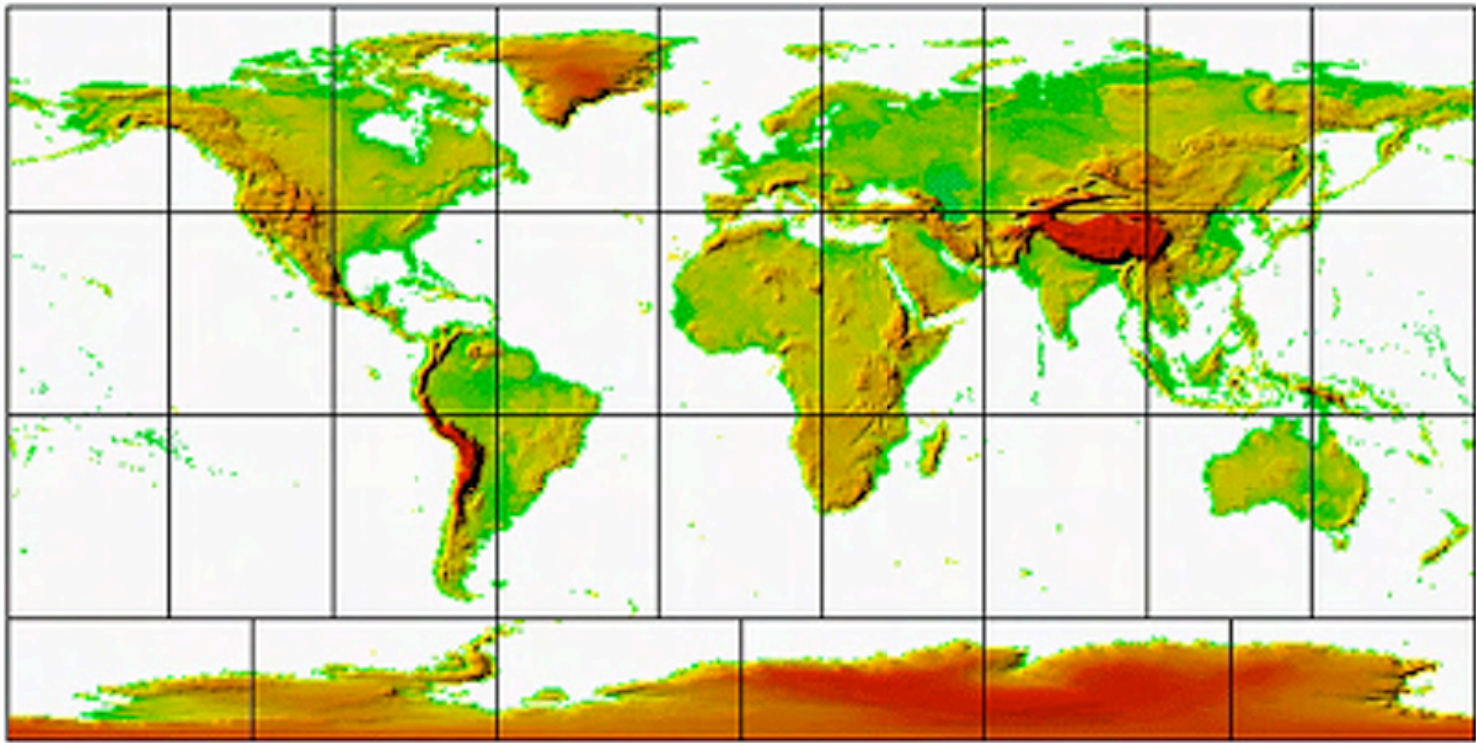
Science

NASA LP DAAC

Satellite Missions

NSLRSDA

About EROS



Click on an area of the map to preview or download data.

Geospatial Domain

What's so special about spatial?

- Point, Line, Polygon, Multi*
 - Raster Data
 - Spatial Reference System (SRS), SRIDs
 - Spatial Indices
 - Map Projections
- 

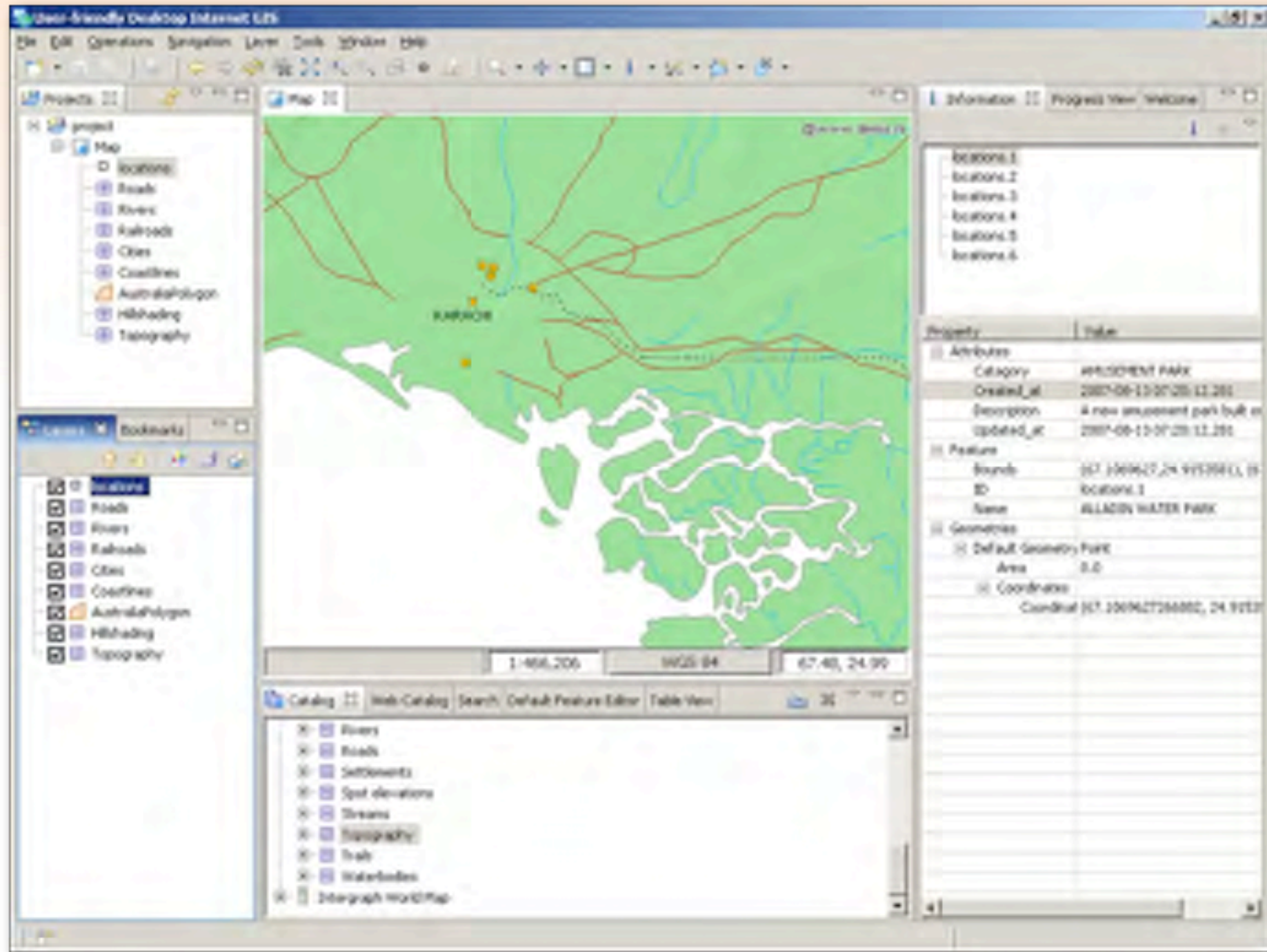
Free desktop GIS (uDig)

<http://udig.refractions.net>



Free desktop GIS (uDig)

<http://udig.refractions.net>



Spatial Adapter (database connection)

```
$ rails railsconfeu07_gis  
$ cd railsconfeu07_gis/
```

```
$ ruby script/plugin install  
svn://rubyforge.org/var/svn/georuby/SpatialAdapter/  
trunk/spatial_adapter
```

```
$ createdb -O sab -T template_postgis  
railsconfeu07_gis_development
```

```
$ createdb -O sab -T template_postgis  
railsconfeu07_gis_test
```


RESTful location

```
$ ruby script/generate scaffold_resource Location
```

geom	:point
name	:string
category	:string
description	:text



Migrations with spatial data

```
def self.up
  create_table :locations do |t|
    t.column :geom, :point, :null => false,
             :srid => 4326, :with_z => true
    t.column :name, :string, :null => false
    t.column :category, :string, :null => false
    t.column :description, :text
  end
end
```

SRID explained

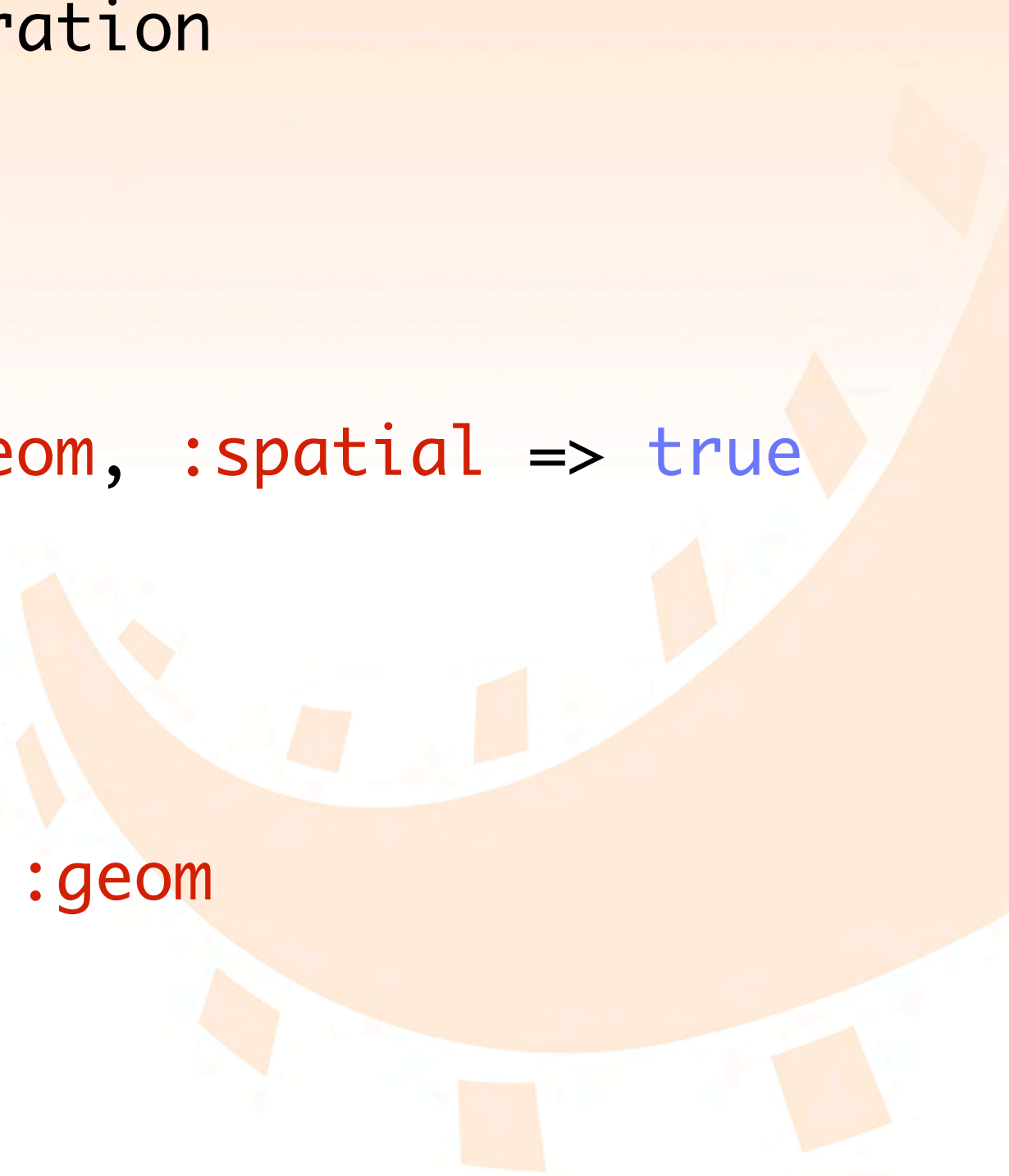
```
$ psql -d template_postgis
template_postgis=# \x -- to turn on expanded display
template_postgis=# SELECT * from spatial_ref_sys where srid = 4326;
-[ RECORD 1 ]-----
srid          | 4326
auth_name     | EPSG
auth_srid     | 4326
srttext       | GEOGCS["WGS 84",DATUM["WGS_1984",
    SPHEROID["WGS 84",6378137,298.25722 3563, AUTHORITY["EPSG","7030"]],
    TOWGS84[0,0,0,0,0,0,0], AUTHORITY["EPSG","6326"]],
    PRIMEM["Greenwich",0, AUTHORITY["EPSG","8901"]],
    UNIT["degree",0.01745329251994328, AUTHORITY["EPSG","9122"]],
    AUTHORITY["EPSG","4326"]]
proj4text     | +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs
```

Spatial indices

```
$ ruby script/generate migration  
add_index_to_locations
```

```
def self.up  
  add_index :locations, :geom, :spatial => true  
end
```

```
def self.down  
  remove_index :locations, :geom  
end
```



Spatial data migrations (points)

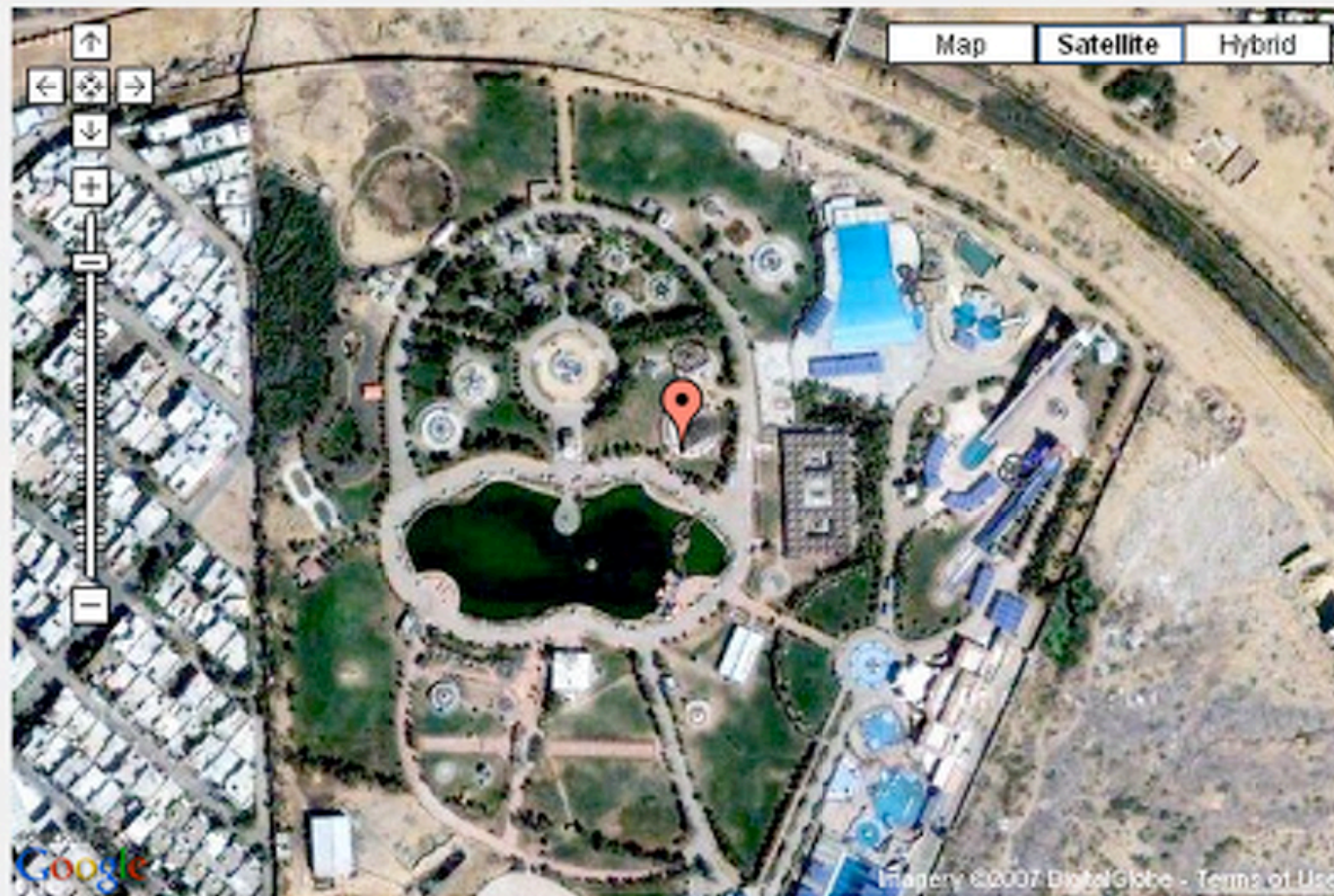
```
$ ruby script/generate migration add_locations_data
```

```
def self.up
  Location.create
    :geom          => Point.from_x_y_z( 67.1069627266882,
                                         24.9153581895111,
                                         3,
                                         4326 ),
    :name          => "ALLADIN WATER PARK",
    :category      => "AMUSEMENT PARK",
    :description   => "A new amusement park built on
                      the main Rashid Minhas Road is
                      the latest attraction of Karachi..."
end
```

<http://localhost:3000/locations/1>

ALLADIN WATER PARK

Geom:



Category: AMUSEMENT PARK

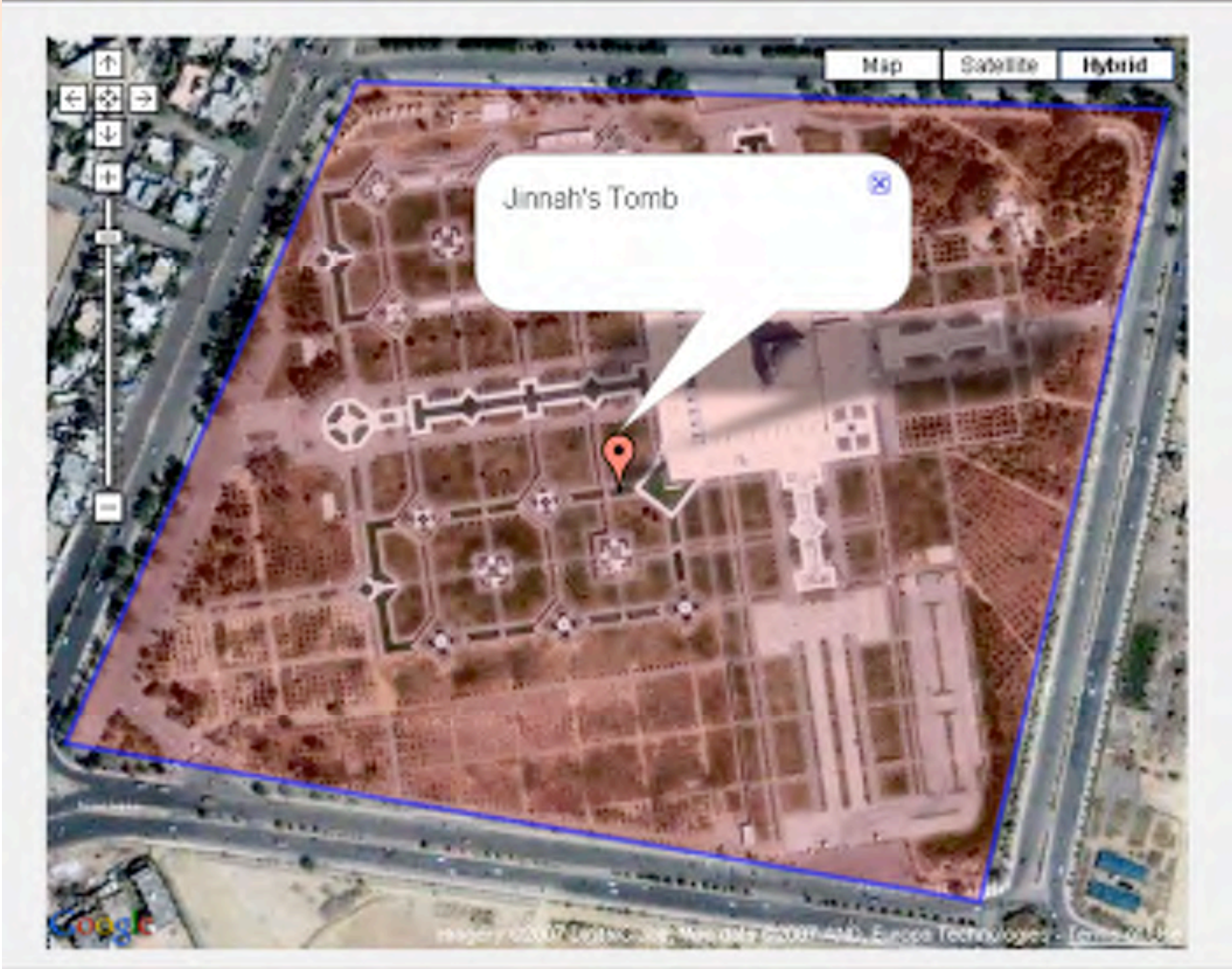
Description: A new amusement park built on the main Rashid Minhas Road is the latest attraction of Karachi. It has the colorful slides, one of the tallest in Asia. It is

Spatial data migrations (polygon)

```
loc = Location.new( :name => "Jinnah's Tomb",  
  :geom => Polygon.from_coordinates(  
    [[  
      [67.04331636428833, 24.87680084533657],  
      [67.03837037086487, 24.876956578914644],  
      [67.03658938407898, 24.87328705430993],  
      [67.04217910766602, 24.87241102436732],  
      [67.04331636428833, 24.87680084533657]  
    ]], 3, 4326)  
)
```




Jinnah's Tomb



Tests (fixtures)

one:

id: 1

geom: `<%= Point.from_x_y_z(67.0665783392958,
24.9357149717549,
3,
4326
)>.to_fixture_format %>`

name: "GULBERG POLICE STATION"

category: "POLICE"

description: "Yet another well regarded police
station in Karach"

Tests (unit/functional)

```
def test_should_create_location
  old_count = Location.count
  post :create, :location => {
    :geom => Point.from_x_y_z(67.0665783392958,
                              24.9357149717549, 3, 4326),
    :name      => "GULBERG POLICE STATION",
    :category   => "POLICE",
    :description => "Yet another well regarded police
                    station in Karachi"
  }
  assert_equal old_count+1, Location.count
  assert_redirected_to location_path( assigns(:location) )
end
```


Tests (unit/functional)



\$ rake test:units

\$ rake test:functionals

Tests (JavaScript)

```
$ ruby script/plugin install
```

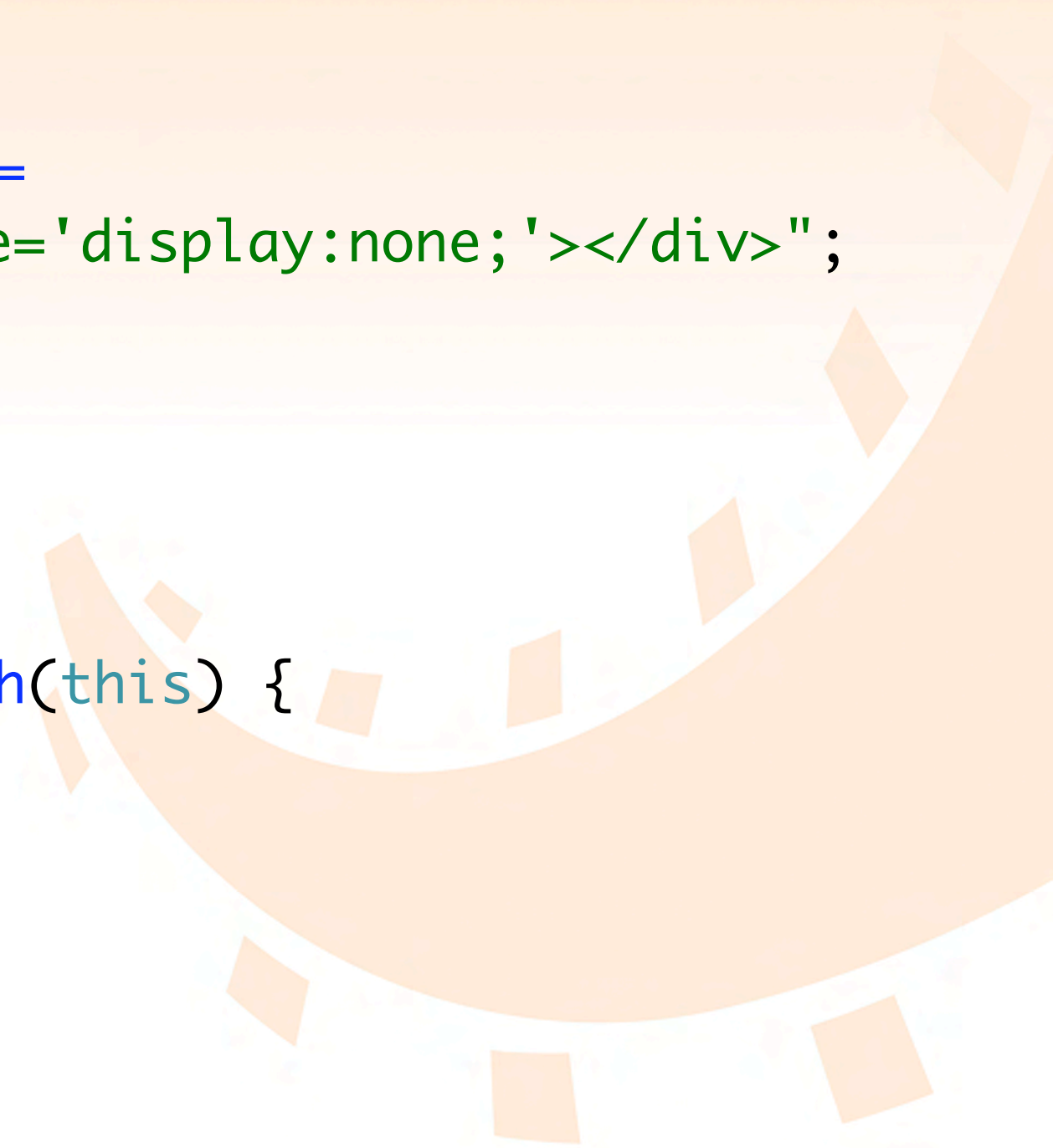
```
http://dev.rubyonrails.org/svn/rails/plugins/  
javascript\_test
```

```
$ ruby script/generate javascript_test application
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has a thick orange border and several rectangular teeth protruding from its outer edge. The gear is partially cut off by the right edge of the slide.

Tests (JavaScript)

```
new Test.Unit.Runner({  
  
  setup: function() {  
    $('sandbox').innerHTML =  
      "<div id='123_a' style='display:none;'></div>";  
  },  
  
  teardown: function() {  
  },  
  
  testTruth: function() { with(this) {  
    assert(true);  
  }}  
  
}, "testlog");
```



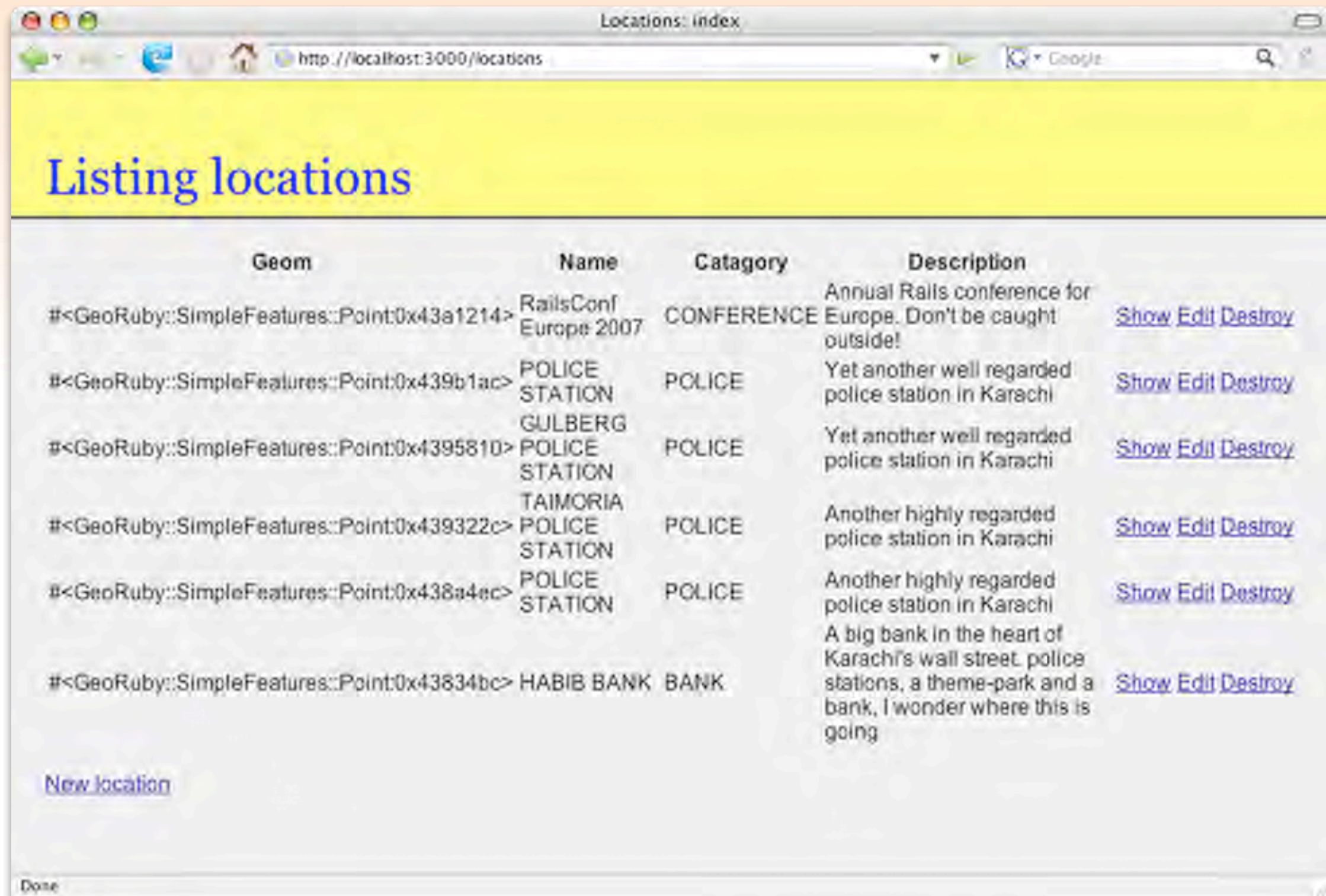
Tests (JavaScript)



```
$ rake test:javascripts
```

GeoRails Stack

What do we have so far?



The screenshot shows a web browser window titled 'Locations: index' with the URL 'http://localhost:3000/locations'. The page has a yellow header with the text 'Listing locations'. Below the header is a table with four columns: 'Geom', 'Name', 'Category', and 'Description'. The table contains six rows of data. Each row has a 'Show Edit Destroy' link to its right. At the bottom left of the table area is a link 'New location'. The browser's status bar at the bottom says 'Done'.

Geom	Name	Category	Description	
#<GeoRuby::SimpleFeatures::Point:0x43a1214>	RailsConf Europe 2007	CONFERENCE	Annual Rails conference for Europe. Don't be caught outside!	Show Edit Destroy
#<GeoRuby::SimpleFeatures::Point:0x439b1ac>	POLICE STATION	POLICE	Yet another well regarded police station in Karachi	Show Edit Destroy
#<GeoRuby::SimpleFeatures::Point:0x4395810>	GULBERG POLICE STATION	POLICE	Yet another well regarded police station in Karachi	Show Edit Destroy
#<GeoRuby::SimpleFeatures::Point:0x439322c>	TAIMORIA POLICE STATION	POLICE	Another highly regarded police station in Karachi	Show Edit Destroy
#<GeoRuby::SimpleFeatures::Point:0x438a4ec>	POLICE STATION	POLICE	Another highly regarded police station in Karachi	Show Edit Destroy
#<GeoRuby::SimpleFeatures::Point:0x43834bc>	HABIB BANK	BANK	A big bank in the heart of Karachi's wall street. police stations, a theme-park and a bank, I wonder where this is going	Show Edit Destroy

[New location](#)

GeoRails Stack: YM4R

```
$ ruby script/plugin install  
svn://rubyforge.org/var/svn/ym4r/Plugins/GM/  
trunk/ym4r_gm
```

```
$RAILS_ROOT/config/gmaps_api_key.yml
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has several teeth and a circular center, rendered in a light orange color that matches the slide's background.

GeoRails Stack

The complete stack

YM4R

Spatial Adapter

GeoRuby

PostGIS Database

CRUD location: show (helper)

```
def show_map rec
  @map = GMap.new( "map#{rec.id}_div" )
  @map.control_init :large_map => true,
                   :map_type => true

  @map.center_zoom_init( [rec.geom.y,rec.geom.x], 16 )
  @map.set_map_type_init( GMapType::G_SATELLITE_MAP )
  @map.overlay_init( GMarker.new([rec.geom.y,rec.geom.x],
                                :title => rec.name,
                                :info_window => rec.description))
end
```

CRUD location: index (helper)

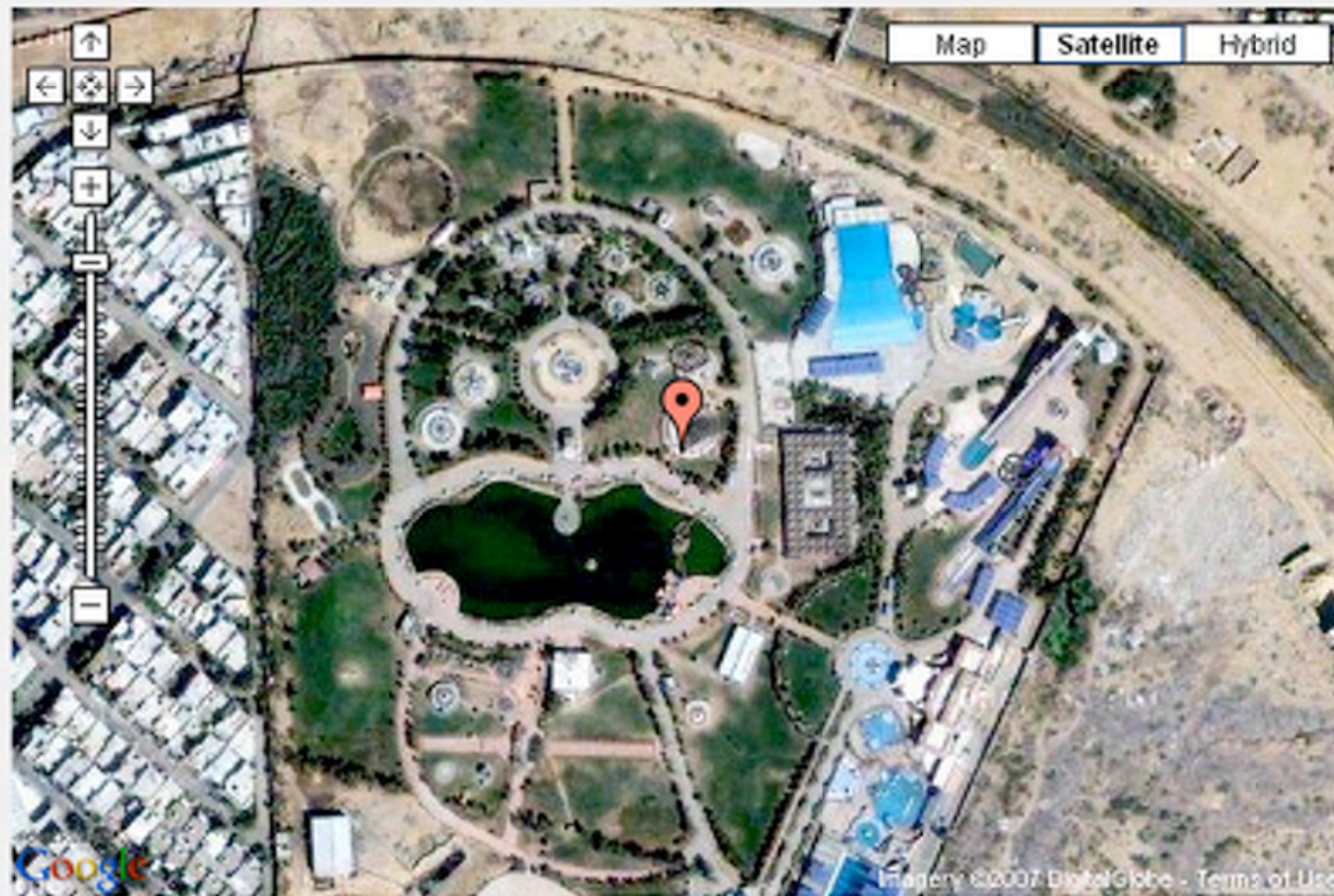
```
def show_maps recs
  @maps = Hash.new
  recs.each do |rec|
    map = GMap.new("map#{rec.id}_div", "map#{rec.id}")
    map.control_init(:small_map => true, :map_type => true)
    map.center_zoom_init([rec.geom.y, rec.geom.x], 16)
    map.set_map_type_init(GMapType::G_SATELLITE_MAP)
    map.overlay_init(GMarker.new([rec.geom.y, rec.geom.x],
      :title => rec.name,
      :info_window => rec.catagory))

    @maps[rec.id] = map
  end
end
```


<http://localhost:3000/locations/1>

ALLADIN WATER PARK

Geom:



Category: AMUSEMENT PARK

Description: A new amusement park built on the main Rashid Minhas Road is the latest attraction of Karachi. It has the colorful slides, one of the tallest in Asia. It is

CRUD location: index (view template)

```
def show_maps recs
  @maps = Hash.new
  recs.each do |rec|
    map = GMap.new("map#{rec.id}_div", "map#{rec.id}")
    map.control_init(:small_map => true, :map_type => true)
    map.center_zoom_init([rec.geom.y, rec.geom.x], 16)
    map.set_map_type_init(GMapType::G_SATELLITE_MAP)
    map.overlay_init(GMarker.new([rec.geom.y, rec.geom.x],
      :title => rec.name,
      :info_window => rec.catagory))

    @maps[rec.id] = map
  end
end
```


CRUD location: index (view template)

```
<%= GMap.header %>
<% show_maps @locations %>
<% @maps.each_value do |map| %>
  <%= map.to_html %>
<% end %>
...
<% for location in @locations %>
  <%= @maps[location.id].div :width => 240,
                             :height => 160 %>
<% end %>
```


<http://localhost:3000/locations>

Listing locations

Geom	Name	Category	Description
	RailsConf Europe 2007	CONFERENCE	Annual Rails conference for Europe. Don't be caught outside! Show Edit Destroy
	POLICE STATION	POLICE	Yet another well regarded police station in Karachi Show Edit Destroy
	GULBERG POLICE STATION	POLICE	Yet another well regarded police station in Karachi Show Edit Destroy

CRUD location: new (JavaScript application.js)

```
function create_draggable_editable_marker()
{
    // initialize the values in form fields to 0
    document.getElementById("lng").value = 0;
    document.getElementById("lat").value = 0;
    var currMarker;

    // if the user click on an existing marker remove it
    GEvent.addListener(map, "click", function(marker, point) {
    if (marker) {
        if (confirm("Do you want to delete marker?")) {
            map.removeOverlay(marker);
        }
    }
    }
}
```

CRUD location: new (JavaScript application.js)

```
// if the user clicks somewhere other than existing marker
else {
    // remove the previous marker if it exists
    if (currMarker) { map.removeOverlay(currMarker); }
    currMarker = new GMarker(point, {draggable: true});
    map.addOverlay(currMarker);

    // update the form fields
    document.getElementById("lng").value = point.x;
    document.getElementById("lat").value = point.y;
}
// Similarly drag event is used to update the form fields
GEvent.addListener(currMarker, "drag", function() {
    document.getElementById("lng").value = currMarker.getPoint().lng();
    document.getElementById("lat").value = currMarker.getPoint().lat();
});
});
}
```

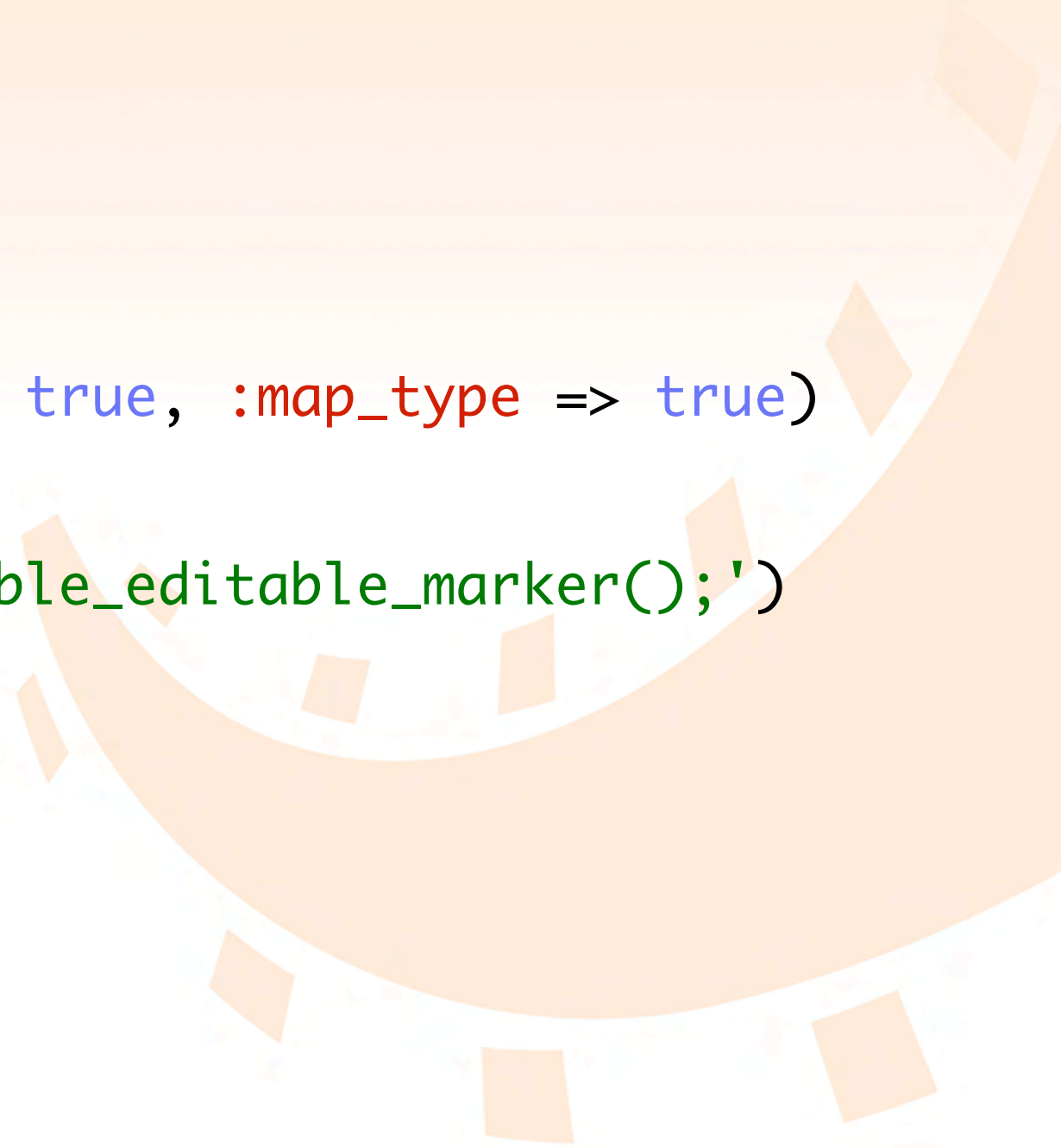

CRUD location: new (layouts/locations.rb)

```
<%= javascript_include_tag :defaults %>
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. The gear has a thick orange body and a series of rectangular teeth along its outer edge. The background of the slide is a light orange gradient.

CRUD location: new (helper)

```
def new_map
  @map = GMap.new("map_div")
  @map.control_init(:large_map => true, :map_type => true)
  @map.center_zoom_init([0,0],2)
  @map.record_init('create_draggable_editable_marker();')
end
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has a thick orange body and several rectangular teeth protruding from its outer edge. The gear is partially cut off by the right edge of the frame.

CRUD location: new (view template)

```
<%= GMap.header %>
```

```
<% new_map %>
```

```
<%= @map.to_html %>
```

...

```
<%= @map.div :width => 600, :height => 400 %>
```

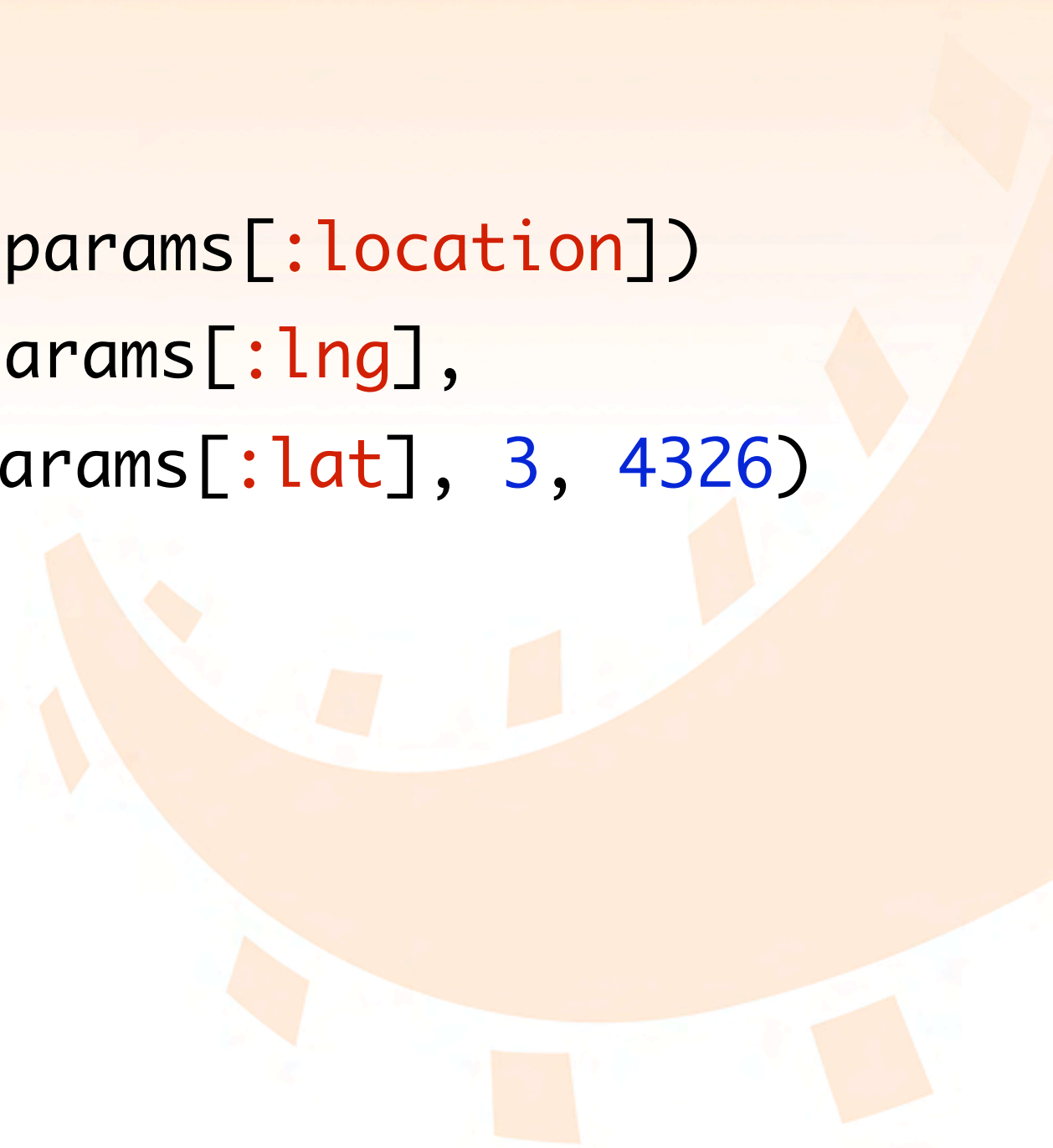
```
Lat: <%= text_field_tag :lat -%>
```

```
Lng: <%= text_field_tag :lng -%>
```


CRUD location: new (controller)

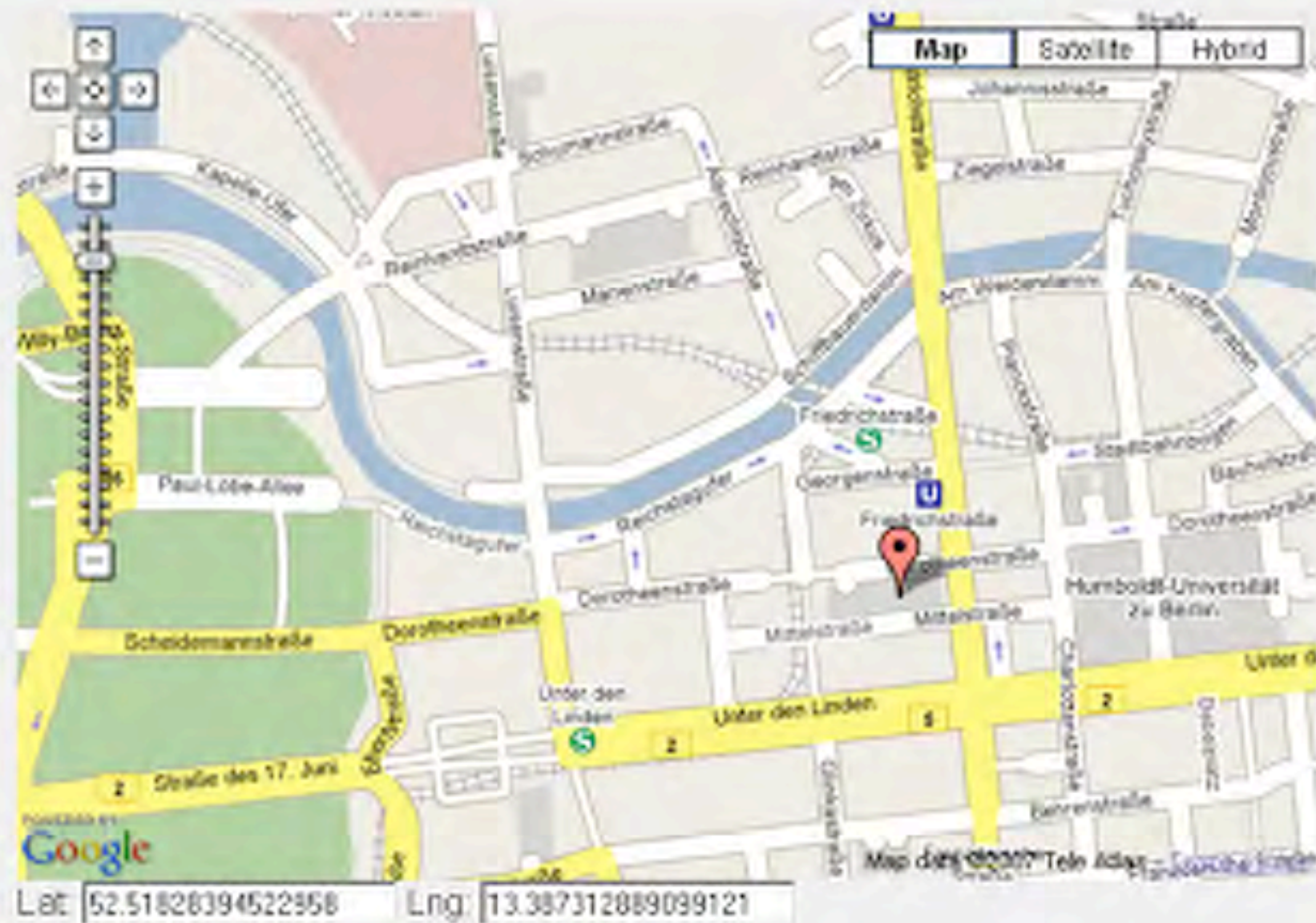
```
def create
  @location = Location.new(params[:location])
  geom = Point.from_x_y_z(params[:lng],
                          params[:lat], 3, 4326)

  @location.geom = geom
  ...
end
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has a thick orange body and several rectangular teeth protruding from its outer edge. The gear is partially cut off by the right edge of the frame.

<http://localhost:3000/locations/new>

New location

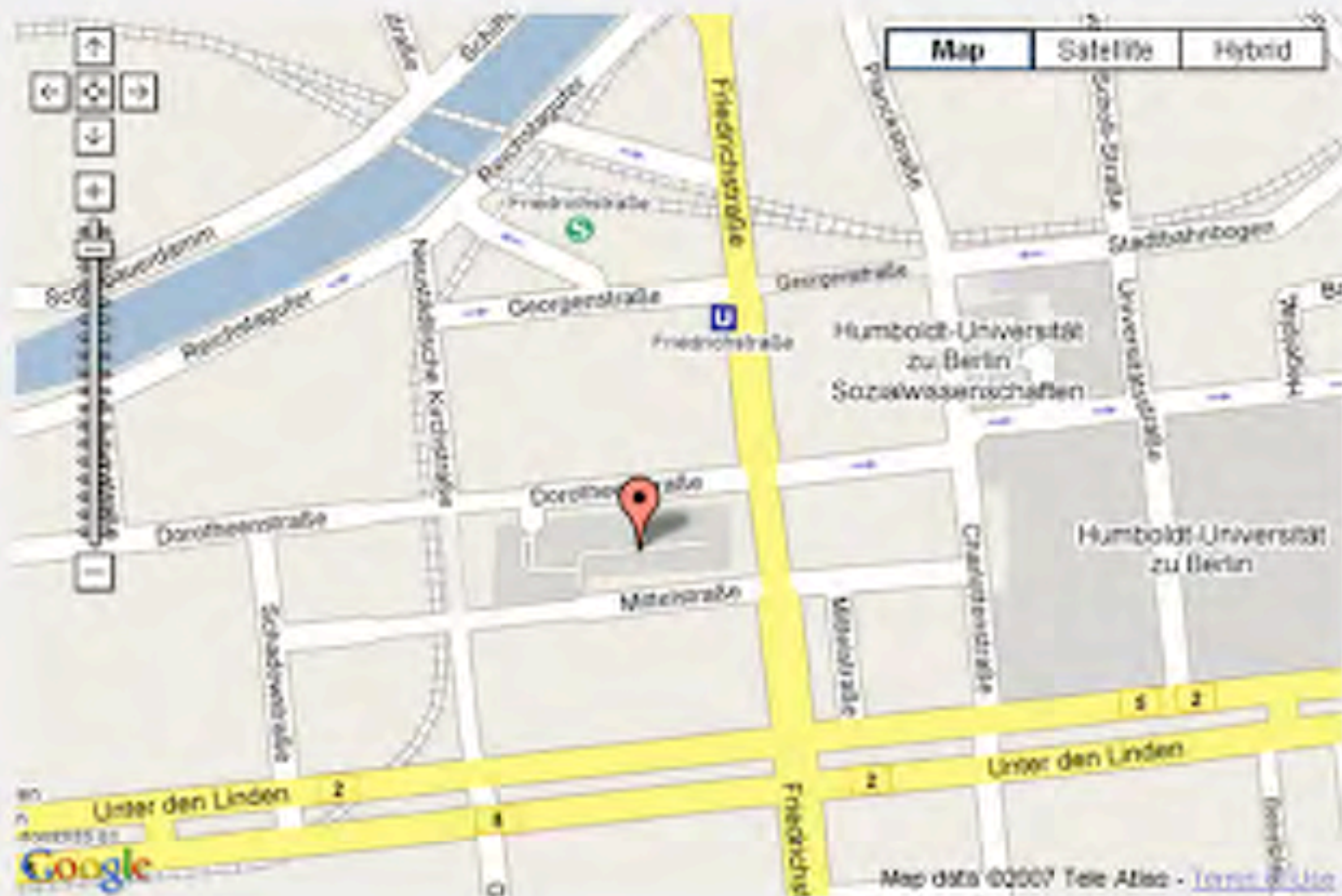


Name

ReisConf Europe 2007

<http://localhost:3000/locations/new>

RailsConf Europe 2007



Category:

Description:

[Edit](#) | [Back](#)

CRUD location: edit (JavaScript)

```
function create_draggable_marker_for_edit(lng, lat) {  
  // initialize form fields  
  document.getElementById('lng').value = lng;  
  document.getElementById('lat').value = lat;  
  
  // initialize marker  
  var currMarker = new GMarker( new GLatLng(lat, lng),  
                                {draggable: true} );  
  map.addOverlay(currMarker);  
  
  // Handle drag events to update the form text fields  
  GEvent.addListener(currMarker, 'drag', function() {  
    document.getElementById('lng').value = currMarker.getPoint().lng();  
    document.getElementById('lat').value = currMarker.getPoint().lat();  
  });  
}
```

CRUD location: edit (helper)

[illegible]

CRUD location: edit (view template)

```
<%= GMap.header %>
```

```
<% edit_map @location %>
```

```
<%= @map.to_html %>
```

...

```
<%= @map.div :width => 600, :height => 400 %>
```

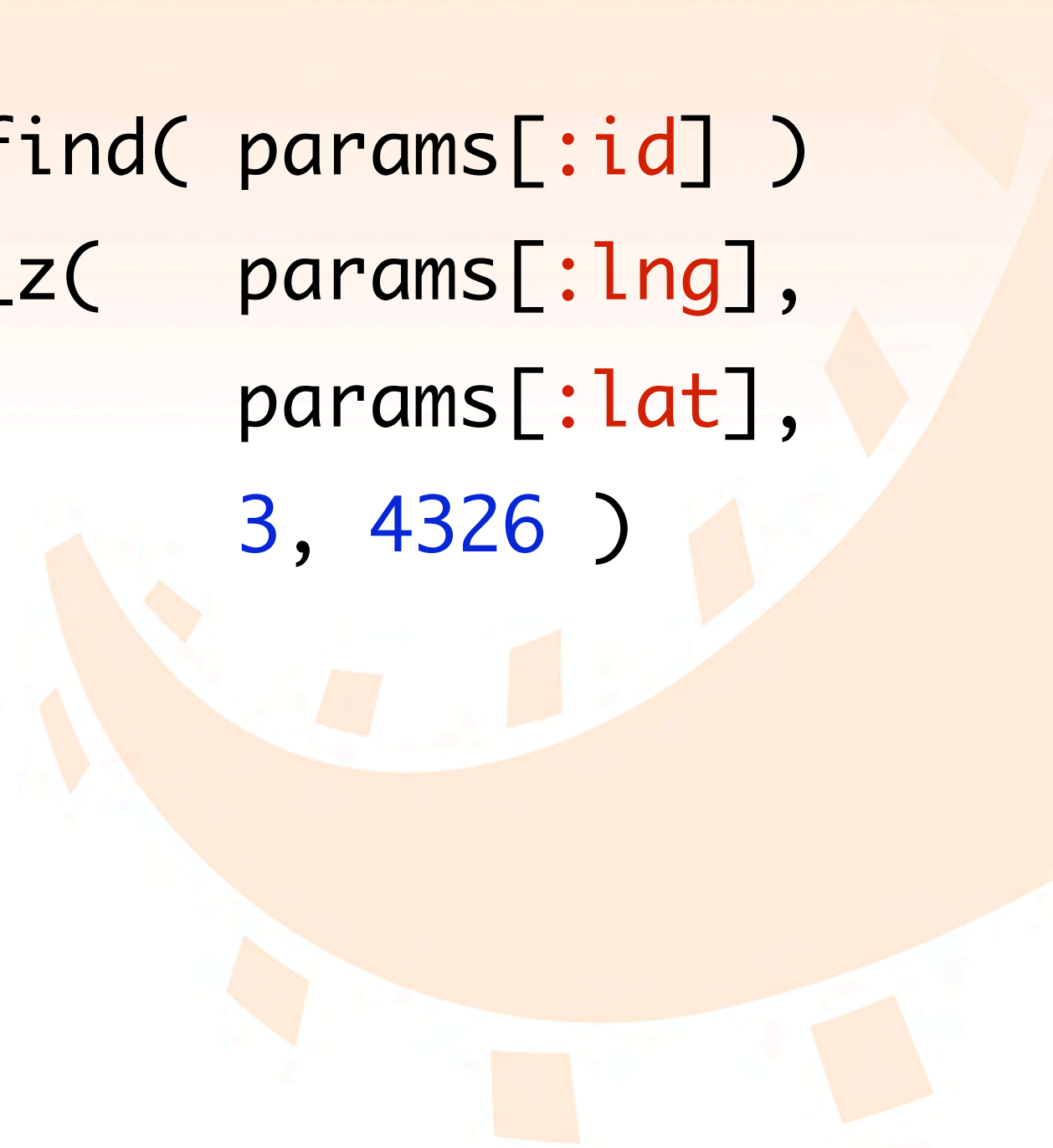
```
Lat: <%= text_field_tag :lat -%>
```

```
Lng: <%= text_field_tag :lng -%>
```


CRUD location: edit (controller)

```
def update
  @location = Location.find( params[:id] )
  geom = Point.from_x_y_z(    params[:lng],
                              params[:lat],
                              3, 4326 )

  @location.geom = geom
  ...
end
```

A large, stylized orange gear graphic is positioned in the bottom right corner of the slide. It has a thick orange body and several rectangular teeth protruding from its outer edge. The gear is partially cut off by the right edge of the frame.

<http://localhost:3000/locations/7;edit>

Editing location

The screenshot shows a Google Maps interface for editing a location. The map displays a street grid with a red pin and a yellow line. The location is labeled "RailsConf Europe 2007". The map includes a search bar, a compass, and a scale bar. The location's coordinates are displayed as Lat: 52.51841451673572 and Lng: 13.385272192001343. The map data is from 2007, and the source is cited as "Map data ©2007 Tele Atlas - Terms of Use".

Map data ©2007 Tele Atlas - [Terms of Use](#)

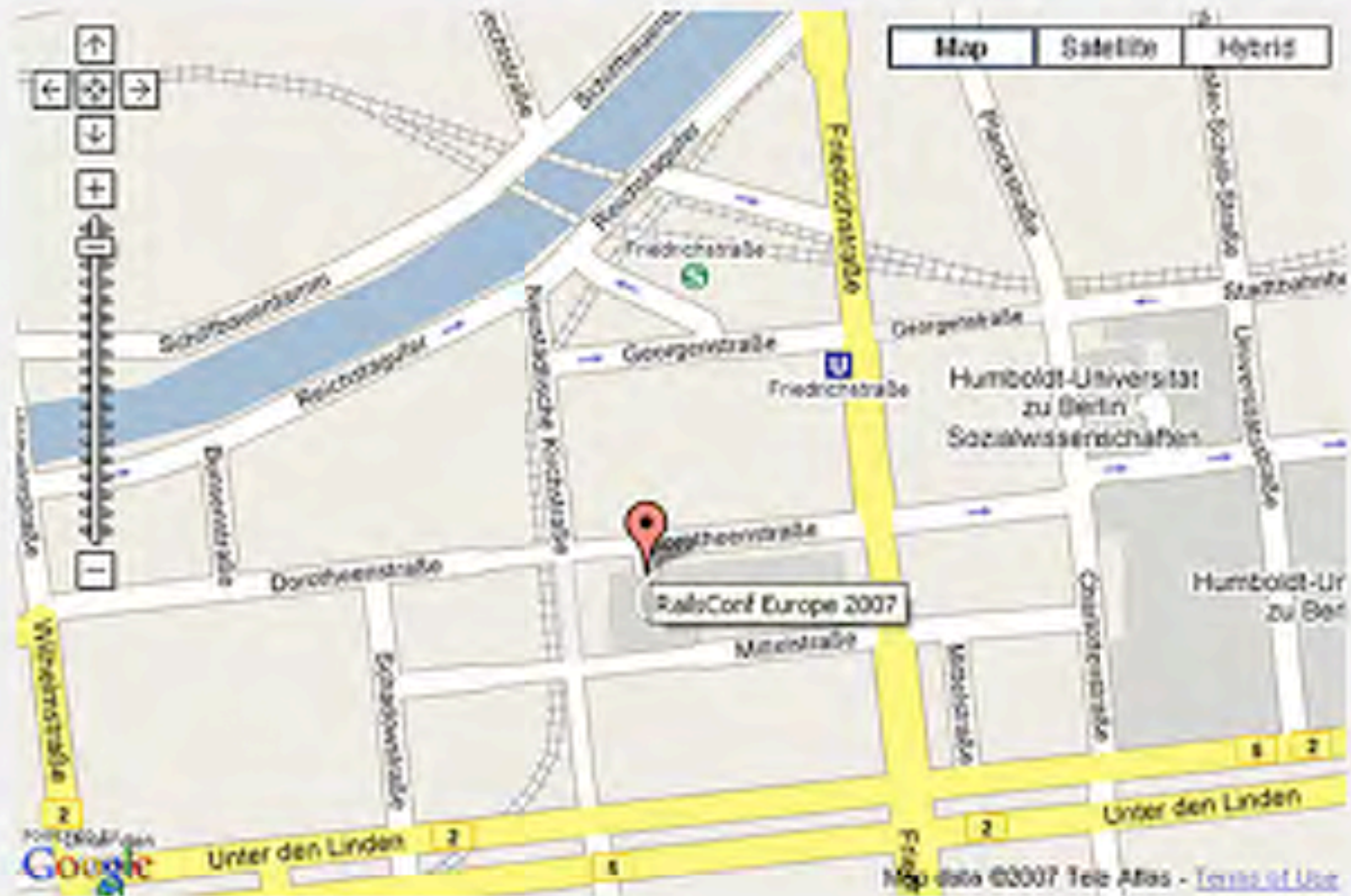
Lat: 52.51841451673572 Lng: 13.385272192001343

Name
RailsConf Europe 2007

Category

<http://localhost:3000/locations/7;edit>

RailsConf Europe 2007



Category: Conference

Description: Annual Rails conference for Europe. Don't miss out!

[Edit](#) | [Back](#)

03

Supporting New Content Types

RESTful Design

Content-type

- KML (Google Earth)
- GeoRSS



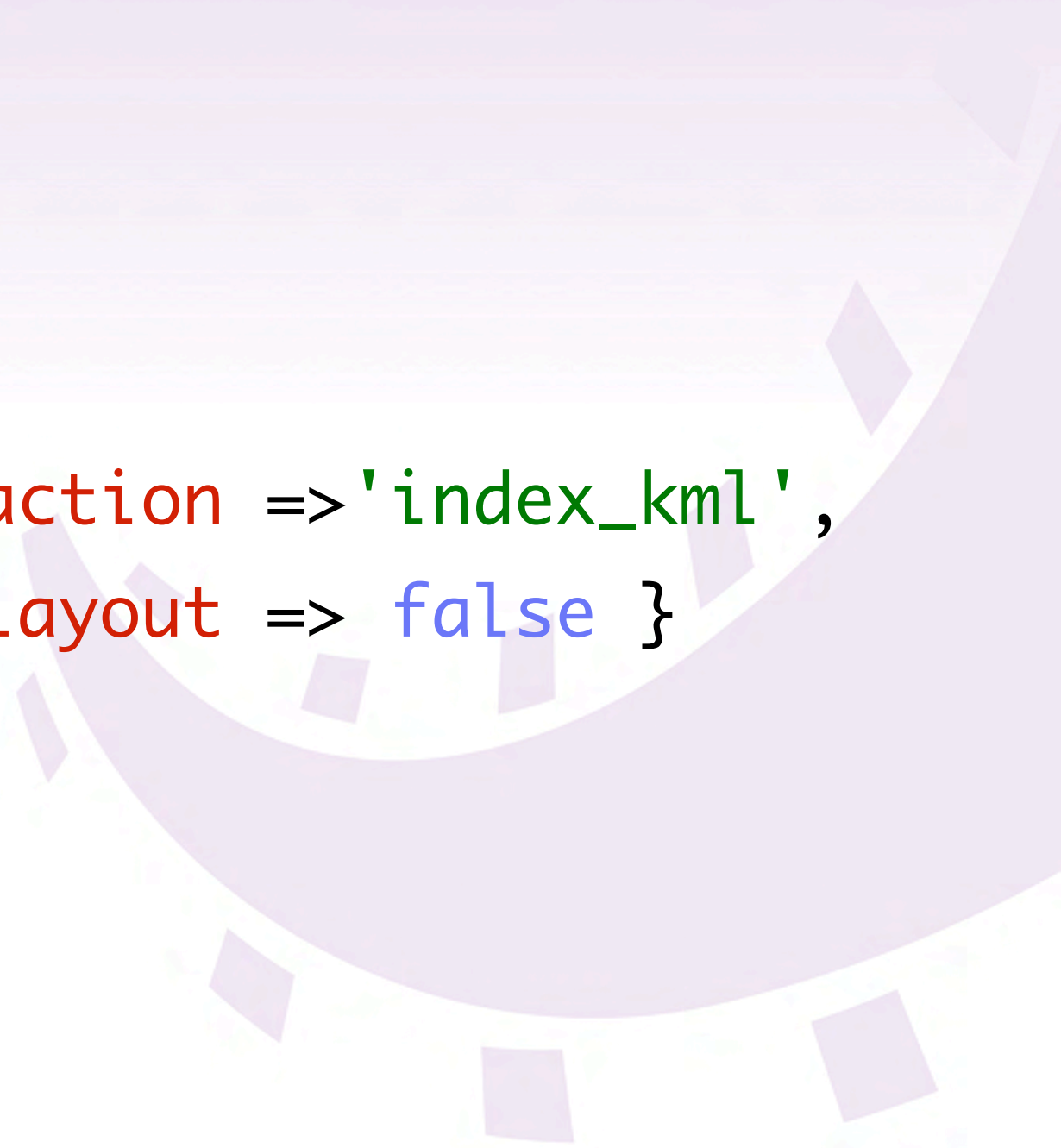
KML – Google Earth (registering new MIME-type)

`Mime::Type.register`

`"application/vnd.google-earth.kml+xml", :kml`

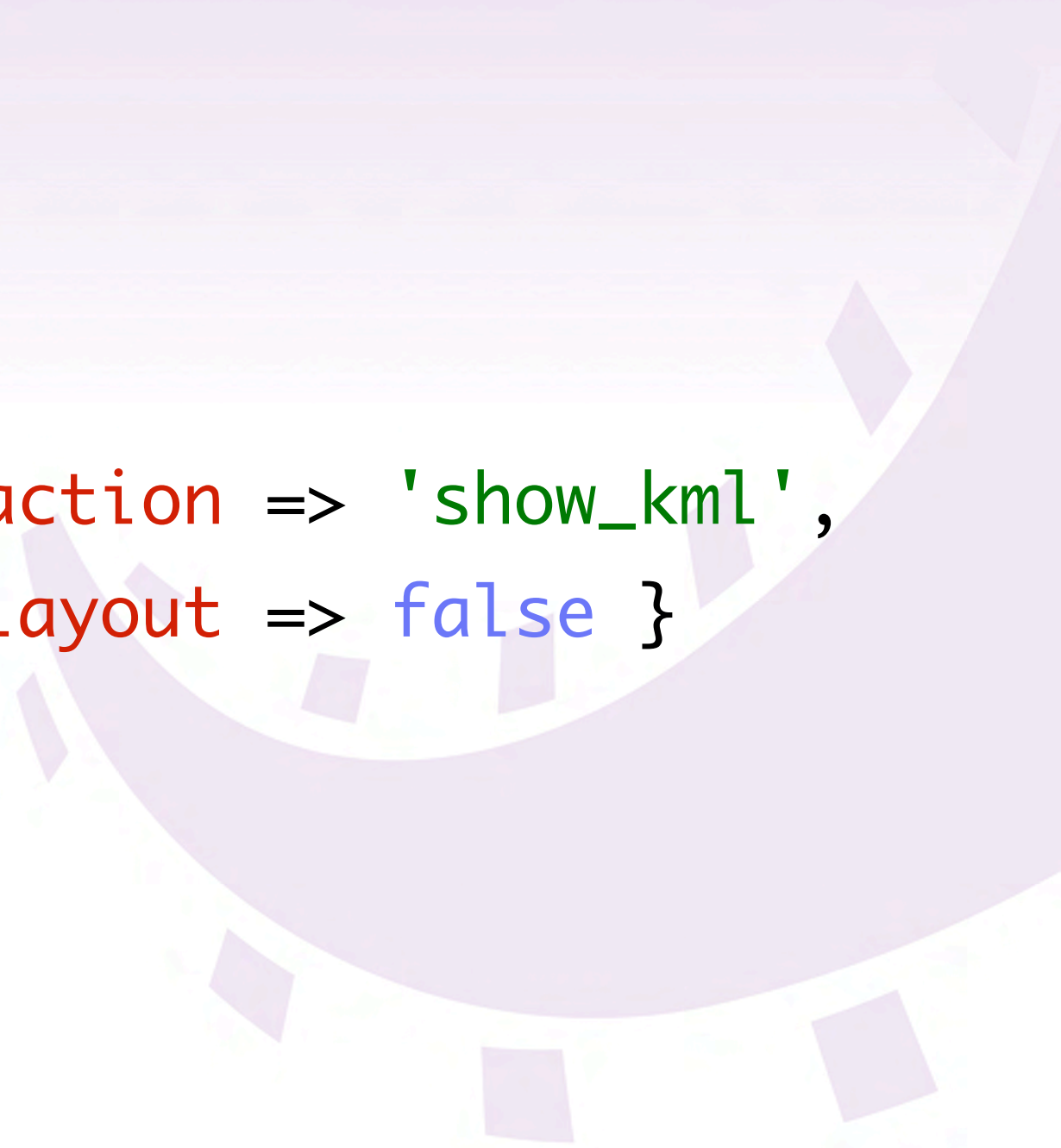
KML – Google Earth (controller)

```
def index
  ...
  respond_to do |format|
    ...
    format.kml { render :action => 'index_kml',
                        :layout => false }
  end
end
```

A decorative background graphic on the right side of the slide. It features a large, light purple gear-like shape. Inside the gear, there is a faint, stylized city skyline with various buildings of different heights. The overall aesthetic is clean and modern, typical of a presentation slide.

KML – Google Earth (controller)

```
def show
  ...
  respond_to do |format|
    ...
    format.kml { render :action => 'show_kml',
                       :layout => false }
  end
end
```



KML – Google Earth (show RXML template)

```
xml.kml("xmlns" => KML_NS) do
  xml.tag! "Document" do
    xml.tag! "Style", :id => "myStyle" do
      xml.tag! "PointStyle" do
        xml.color "ffff0000" #format is aabbggrr
        xml.outline 0
      end
    end
  end
  xml.tag! "Placemark" do
    xml.description @location.description
    xml.name @location.name
    xml.styleUrl "#myStyle"
    xml << @location.geom.as_kml
  end
end
end
```


KML – Google Earth (index RXML template)

```
xml.kml("xmlns" => KML_NS) do
  xml.tag! "Document" do
    xml.tag! "Style", :id => "myStyle" do
      xml.tag! "PointStyle" do
        xml.color "ffff0000" #format is aabbggrr
        xml.outline 0
      end
    end
  end
  @locations.each do |location|
    xml.tag! "Placemark" do
      xml.description location.description
      xml.name location.name
      xml.styleUrl "#myStyle"
      xml << location.geom.as_kml
    end
  end
end
end
```


An aerial photograph of a city, likely London, showing a dense urban landscape with various buildings, streets, and a large railway bridge crossing a river in the lower-left. A white callout box with a blue location pin icon is positioned in the upper-middle section of the image. The callout box contains text about the RailsConf Europe 2007 conference. A yellow pin on the map points to a specific location in the city, which is also labeled with the text 'RailsConf Europe 2007'.

RailsConf Europe 2007

Annual Rails conference for Europe. Don't miss out!

Directions: [To here](#) - [from here](#)

RailsConf Europe 2007

Places

My Places

Signposting

Select this folder and click the 'Play' button below, to

Temporary Places

locations.kml

ALLADIN WATER PARK

A new amusement park built on the main Rashid Minhas Road

POLICE STATION

Yet another well regarded police station in Karachi

QURENG POLICE STATION

Yet another well regarded police station in Karachi

TAMORRA POLICE STATION

Another highly regarded police station in Karachi

POLICE STATION

Another highly regarded police station in Karachi

HABIB BANKS

A big bank in the heart of Karachi's wall street.

Layers

View: Cont

Primary Database

Terrain

Geographic Web

Featured Content

Global Awareness

roads

3D Buildings

borders

Populated Places

Alternative Place Name

Dining

Lodging

Google Earth Community

Shopping and Services

Transportation

Geographic Features

Travel and Tourism

Banks and Finance



KML (formatted URL helpers – index.rhtml)

<h1>

Listing locations

```
<%= link_to image_tag(  
    "/images/google_earth_link.gif"),  
    formatted_locations_path(:kml) %>
```

</h1>

KML (formatted URL helpers – show.rhtml)

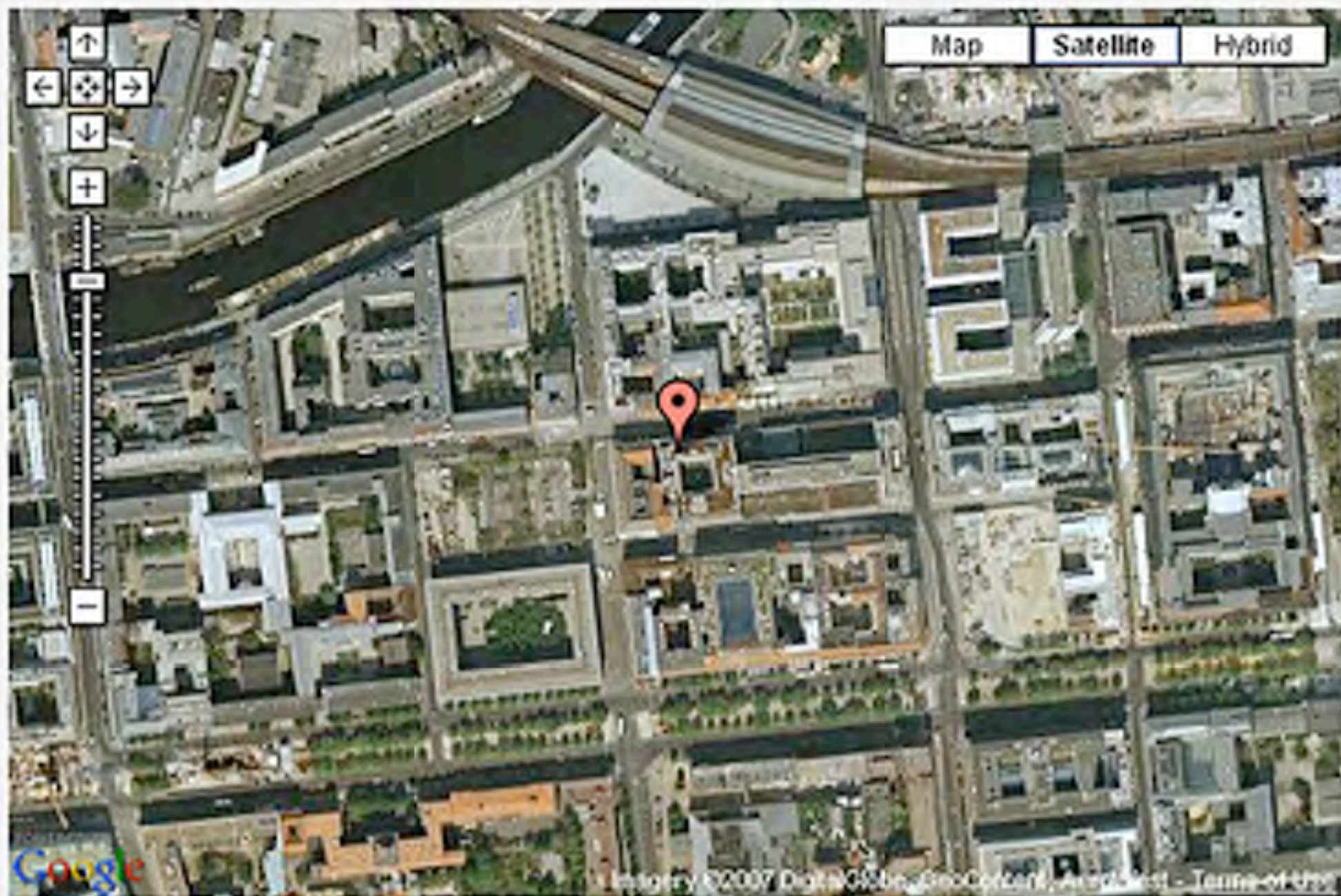
```
<%= link_to image_tag(  
    "/images/google_earth_link.gif"),  
    formatted_location_path(@location, :kml) %>
```

Listing locations

Geom	Name	Catagory	Description
 13.3862721920013 52.5184145167357 3.0	RailsConf Europe 2007	Conference	Annual Rails conference for Europe. Don't miss out! Show Edit Destroy
 149.158458709717 -35.3438350801659 3.0	Geoscience Australia	Government	Australia's National mapping agency. Show Edit Destroy

[New location](#)

RailsConf Europe 2007



GeoRSS (registering new MIME-type)

```
Mime::Type.register "application/georss+xml",  
                    :georss
```



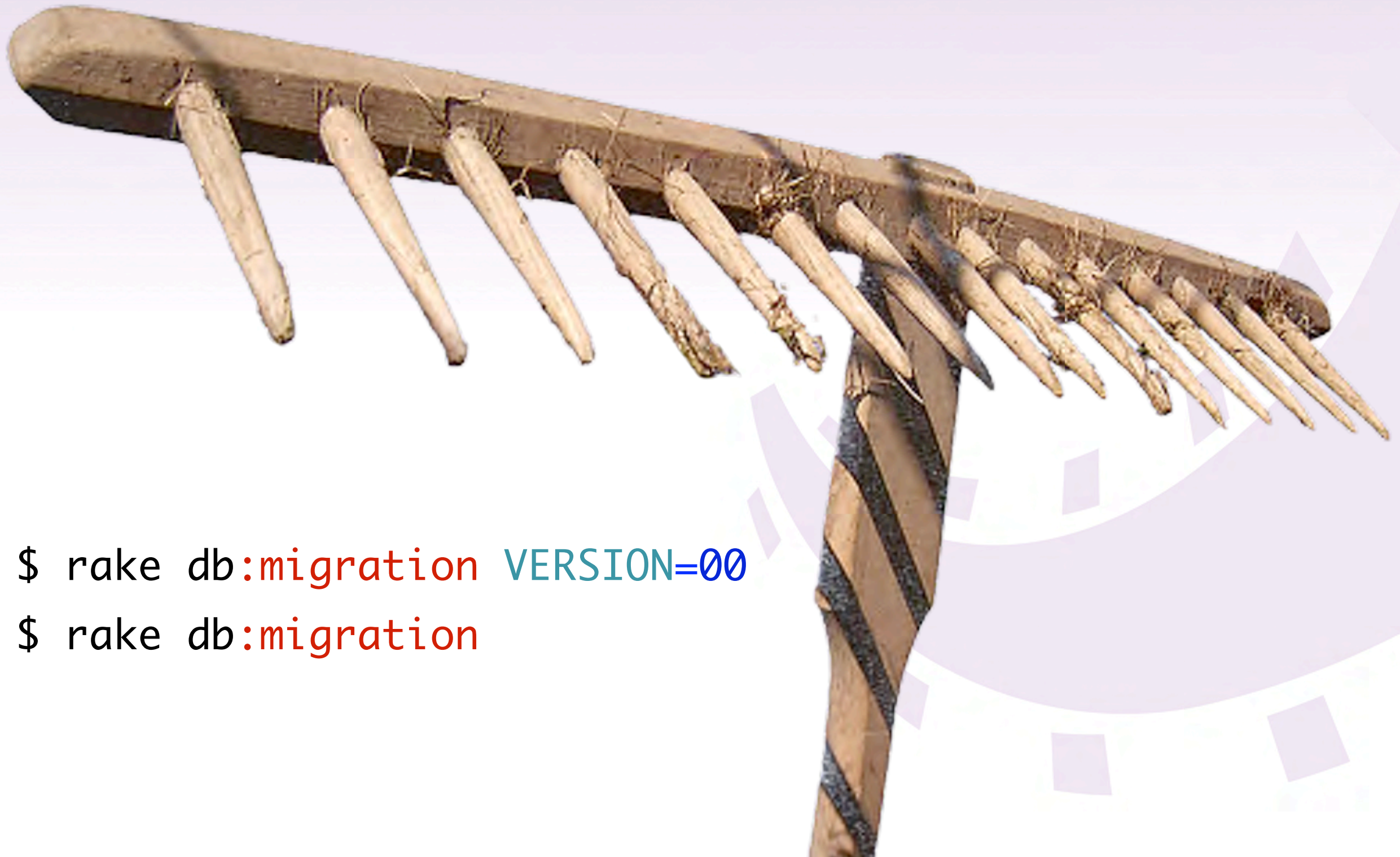
GeoRSS (getting recent locations)

```
$ ruby script/generate migration add_date_fields
```

```
add_column :locations, :created_at, :datetime
```

```
add_column :locations, :updated_at, :datetime
```


GeoRSS (getting recent locations)



```
$ rake db:migration VERSION=00
```

```
$ rake db:migration
```

GeoRSS (model)

```
def self.recent
  self.find :all,
           :limit => 5,
           :order => "updated_at DESC"
end
```



GeoRSS

(controller index action)

```
...  
format.georss {  
  @recent_locations = Location.recent  
  render :action => 'index_georss',  
         :layout => false  
}
```

...

GeoRSS

(controller show action)

```
...  
format.georss {  
  render :action => 'show_georss',  
         :layout => false  
}  
...
```



GeoRSS (show RXML template)

```
xml.rss(:version => "2.0", "xmlns:georss" => GEORSS_NS) do

  xml.channel do
    xml.title "Demo feed for RailsConf Europe 2007"
    xml.link( locations_url )
    xml.description "Coming live from the conf floor!"
    xml.item do
      xml.title @location.name
      xml.link( location_url(@location) )
      xml.description @location.description
      xml << @location.geom.as_georss
    end
  end
end

end
```

GeoRSS (index RXML template)

```
xml.rss(:version => "2.0", "xmlns:georss" => GEORSS_NS) do

  xml.channel do
    xml.title "Demo feed for RailsConf Europe 2007"
    xml.link( locations_url )
    xml.description "Coming live from the conf floor!"
    xml.pubDate( @location.created_at.strftime("%a, %d %b %Y %H:%M:%S %z") )

    @recent_locations.each do |location|
      xml.item do
        xml.title location.name
        xml.link( location_url(location) )
        xml.description location.description
        xml << location.geom.as_georss
      end
    end
  end
end
```




e.g. "10 market st, san francisco" or "hotels near lax"

<http://nomad-labs.dyndns.org/3000/locations.georss>

Search Maps

Search the map

Find businesses

Get directions

Search Results

My Maps

Print Send Link to this page

[Save to My Maps](#)

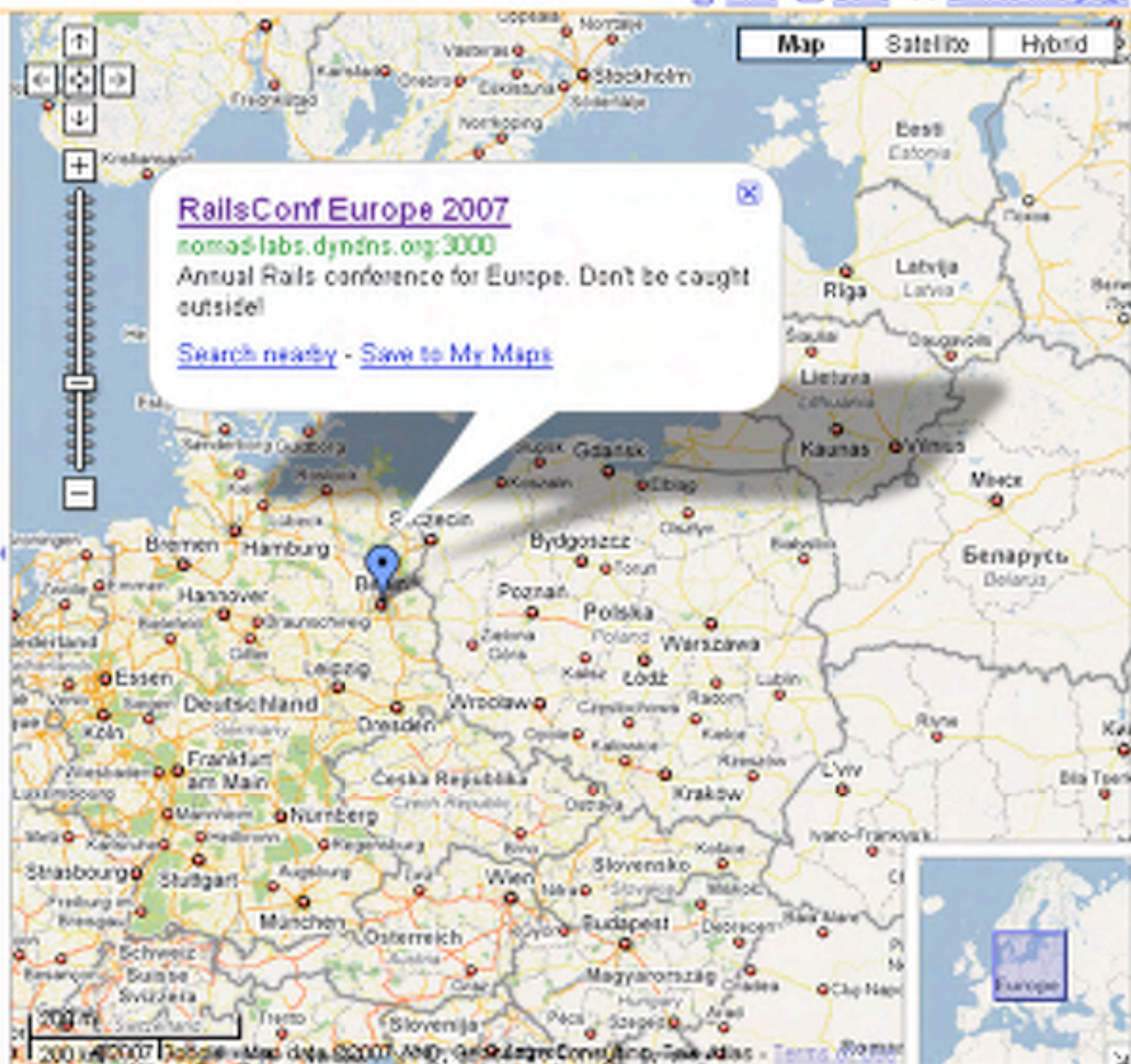
Demo feed for RailsConf Europe 2007

This is only a demo no big deal!

- ☒ [GULBERG POLICE STATION](#)
Yet another well regarded police station in
- ☒ [HABIB BANK](#)
A big bank in the heart of Karachi's wall
- ☒ [POLICE STATION](#)
Another highly regarded police station in
- ☒ [RailsConf Europe 2007](#)
Annual Rails conference for Europe. Don't be
- ☒ [TAMORIA POLICE STATION](#)
Another highly regarded police station in

Displaying content from nomad-labs.dyndns.org

The content overlaid onto this map is provided by a third party, and Google is not responsible for it.



04

Free Libraries

GDAL Ruby Bindings

Credentials

- GDAL is Raster Data IO library
- X/MIT style license
- Google Earth uses it



GDAL

Why do we need it?

- Too many geospatial data formats
- Abstraction layer
- Google Earth uses it!



GDAL

The data model

- Dataset
 - Raster Band
 - Coordinate/Spatial Reference System (CRS/SRS)
 - Affine GeoTransform
 - Ground Control Points (GCP)
- 

A look at the GDAL dataset

```
require 'gdal/gdal'  
require 'gdal/gdalconst'  
include Gdal
```

```
src_file = 'data/bhtmref.img'  
dataset = Gdal::Gdal.open(src_file)
```


GDAL raster band

- Holds the data
- One or more (e.g. RGB, Multi/Hyperspectral)



GDAL raster band

```
dataset.RasterCount
```

```
=> 6
```

```
band = dataset.get_raster_band(1)
```

```
band.XSize
```

```
=> 512
```

```
band.YSize
```

```
=> 512
```



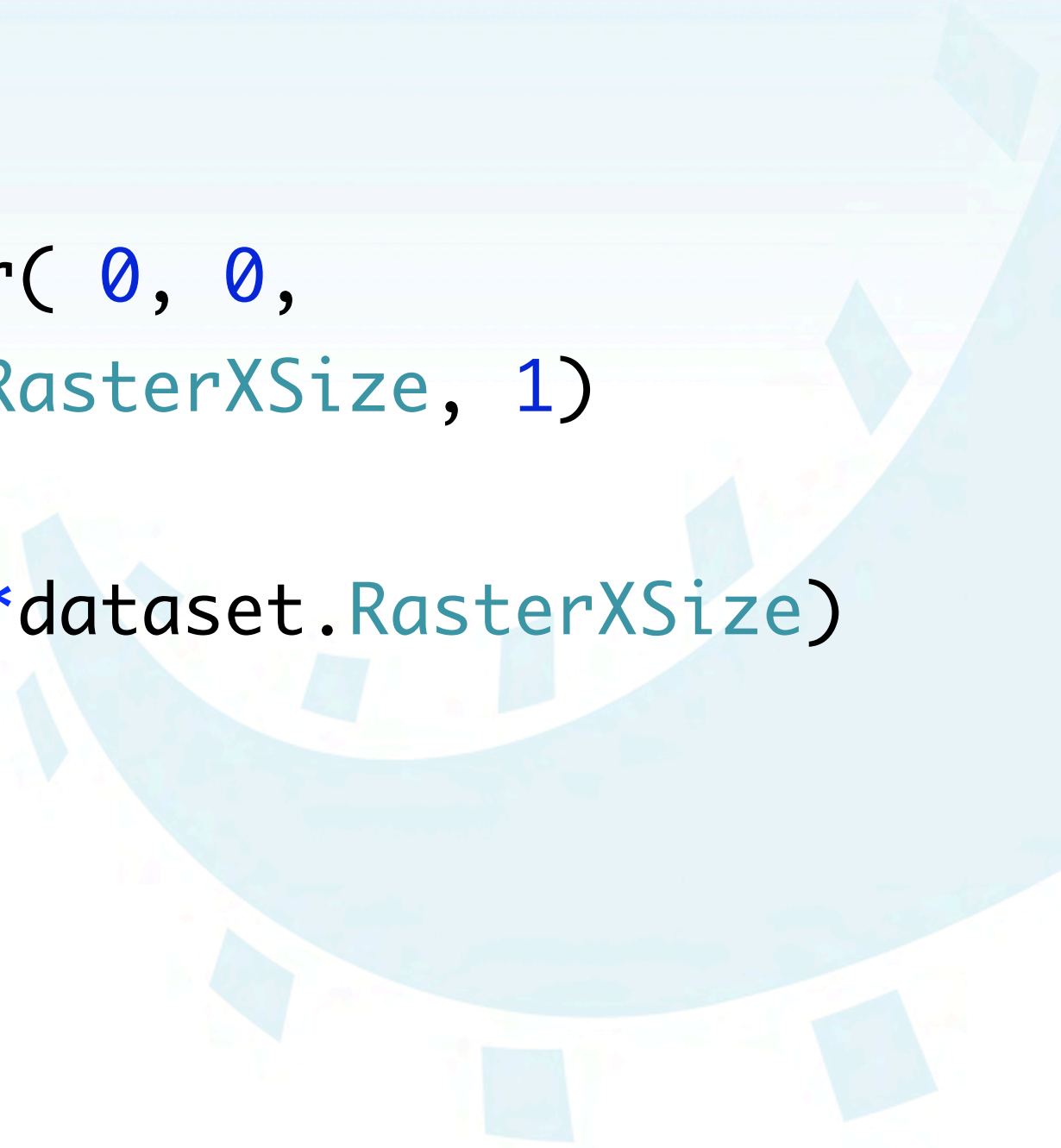
GDAL data type

```
Gdal::Gdal.get_data_type_name(band.DataType)  
=> "Byte"
```



Reading raster data

```
scanline = band.read_raster( 0, 0,  
                             dataset.RasterXSize, 1)  
  
data = scanline.unpack('n'*dataset.RasterXSize)
```



Geo-Transform

```
gt = dataset.get_geo_transform()
```

```
=> [274785.0, 30.0, 0.0, 4906905.0, 0.0, -30.0]
```

gt(0) X Origin (top left corner)

gt(1) X pixel size (W-E pixel resolution)

gt(2) Rotation (0 if north is up)

gt(3) Y Origin (top left corner)

gt(4) Rotation (0 if north is up)

gt(5) -Y pixel size (N-S pixel resolution)

Image-pix to geo-coords

$$X_{\text{geo}} = GT(0) + X_{\text{img}} \times GT(1) + Y_{\text{img}} \times GT(2)$$

$$Y_{\text{geo}} = GT(3) + Y_{\text{img}} \times GT(4) + X_{\text{img}} \times GT(5)$$

gt(0)	X Origin (top left corner)
gt(1)	X pixel size (W-E pixel resolution)
gt(2)	Rotation (0 if north is up)
gt(3)	Y Origin (top left corner)
gt(4)	Rotation (0 if north is up)
gt(5)	−Y pixel size (N-S pixel resolution)

Free library: OSR (spatial reference)

```
require 'gdal/osr'
```

```
srs = Gdal::Osr::SpatialReference.new()
```

```
srs.import_from_wkt(dataset.get_projection)
```

```
=> 0
```

OSR in action

`srs.export_to_pretty_wkt`

```
'PROJCS["UTM Zone 13, Northern Hemisphere",  
  GEOGCS["WGS 84",  
    DATUM["WGS_1984",  
      SPHEROID["WGS 84",6378137,298.257223563,  
        AUTHORITY["EPSG","7030"]],  
      TOWGS84[0,0,0,0,0,0,0],  
      AUTHORITY["EPSG","6326"]],  
    PRIMEM["Greenwich",0,  
      AUTHORITY["EPSG","8901"]],  
    UNIT["degree",0.0174532925199433,  
      AUTHORITY["EPSG","9108"]],  
    AXIS["Lat",NORTH],  
    AXIS["Long",EAST],  
    AUTHORITY["EPSG","4326"]],  
  PROJECTION["Transverse_Mercator"],  
  PARAMETER["latitude_of_origin",0],  
  PARAMETER["central_meridian",-105],  
  PARAMETER["scale_factor",0.9996],  
  PARAMETER["false_easting",500000],  
  PARAMETER["false_northing",0],  
  UNIT["Meter",1]]'
```



Creating Files using GDAL

```
driver = Gdal::Gdal.get_driver_by_name( format )  
metadata = driver.get_metadata_dict
```

```
metadata["DCAP_CREATE"]
```

```
=> "YES"
```

```
metadata["DCAP_CREATECOPY"]
```

```
=> "YES"
```



Creating a copy of dataset

```
src_file = 'data/36q_dtm.tif'
```

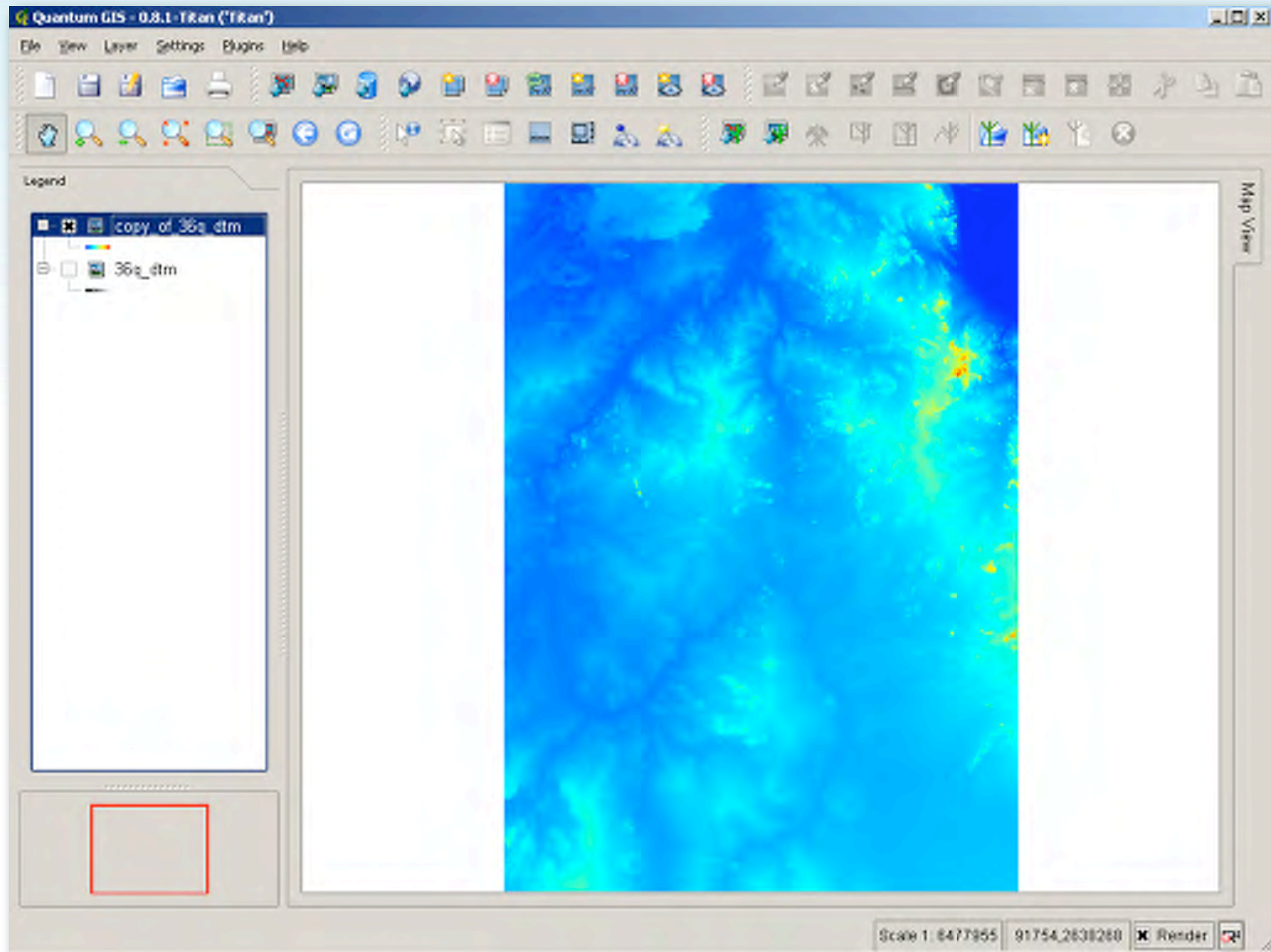
```
dst_file = 'copy_of_36q_dtm.tif'
```

```
src_ds = Gdal::Gdal.open(src_file)
```

```
dst_ds = driver.create_copy( dst_file, src_ds, 0,  
                             [ 'TILED=YES', 'COMPRESS=PACKBITS' ] )
```

```
dst_ds = 0
```

QGIS



nomad

l a b s