

Welcome!

# Geolocation and Maps with PHP

IPC - Mainz, Germany - Oct 10th, 2011

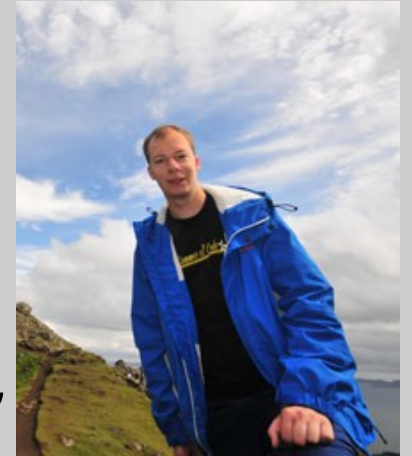
Derick Rethans - [derick@derickrethans.nl](mailto:derick@derickrethans.nl) - twitter:  
@derickr

<http://derickrethans.nl/talks.html>

<http://joind.in/3841>

## Derick Rethans

- Dutchman living in London
- PHP development
- Author of the `mcrypt`, `input_filter`, `dbus`, and date/time extensions
- Author of Xdebug
- Contributor to the Apache Zeta Components Incubator project (formerly eZ Components)
- Freelancer doing PHP (internals) development
- Anderskor



# The Earth is



not a sphere...



... but a bit of a pear.

# The Earth's shape

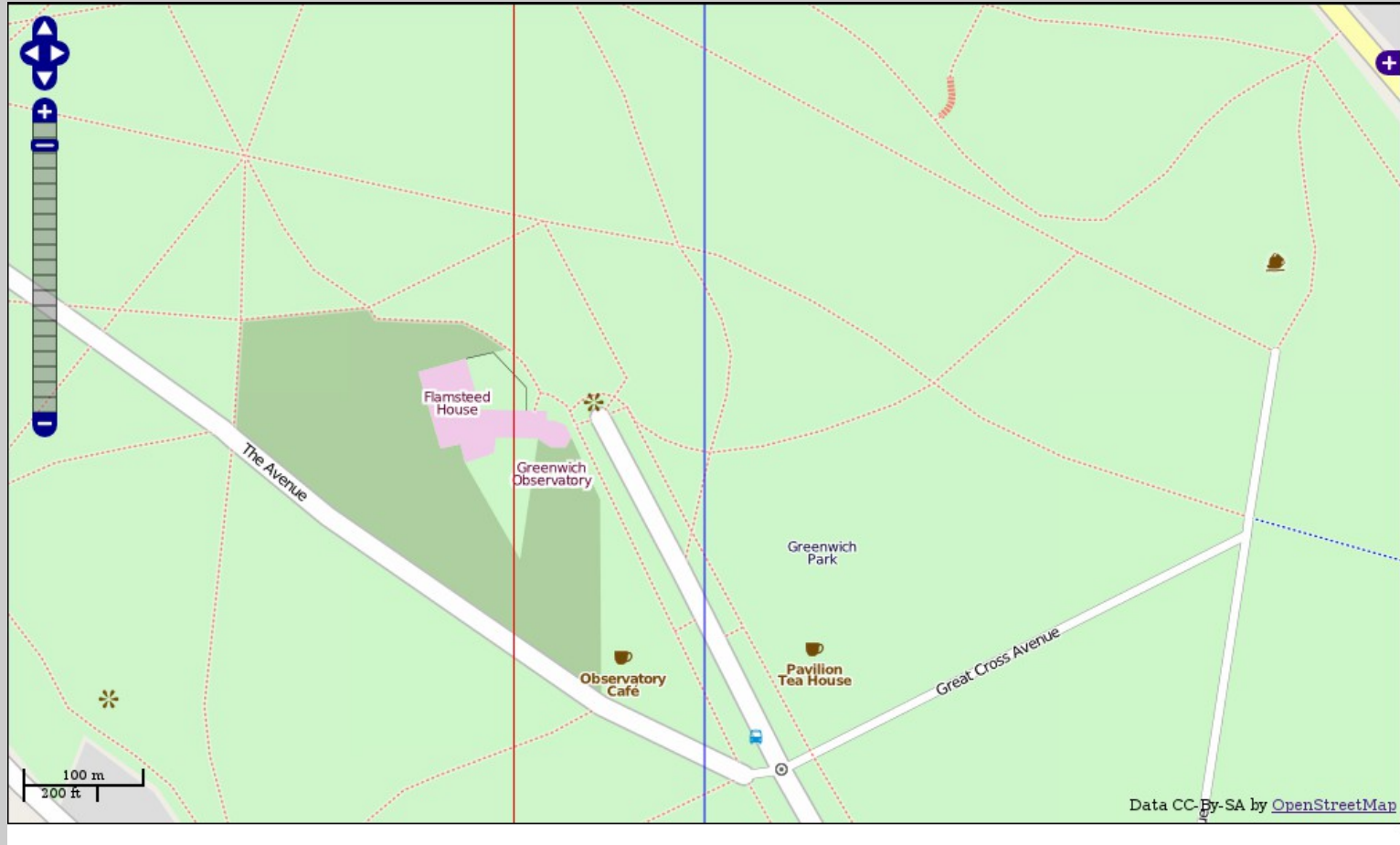
In cartography, the Earth's shape has to be approximated: a reference ellipsoid

- specify the Earth's radius and a flattening constant
- different ones are in use
- also called datum or geodetic system
- for coordinates, a meridian ( $0^\circ$  longitude) should also be specified

Important ones are:

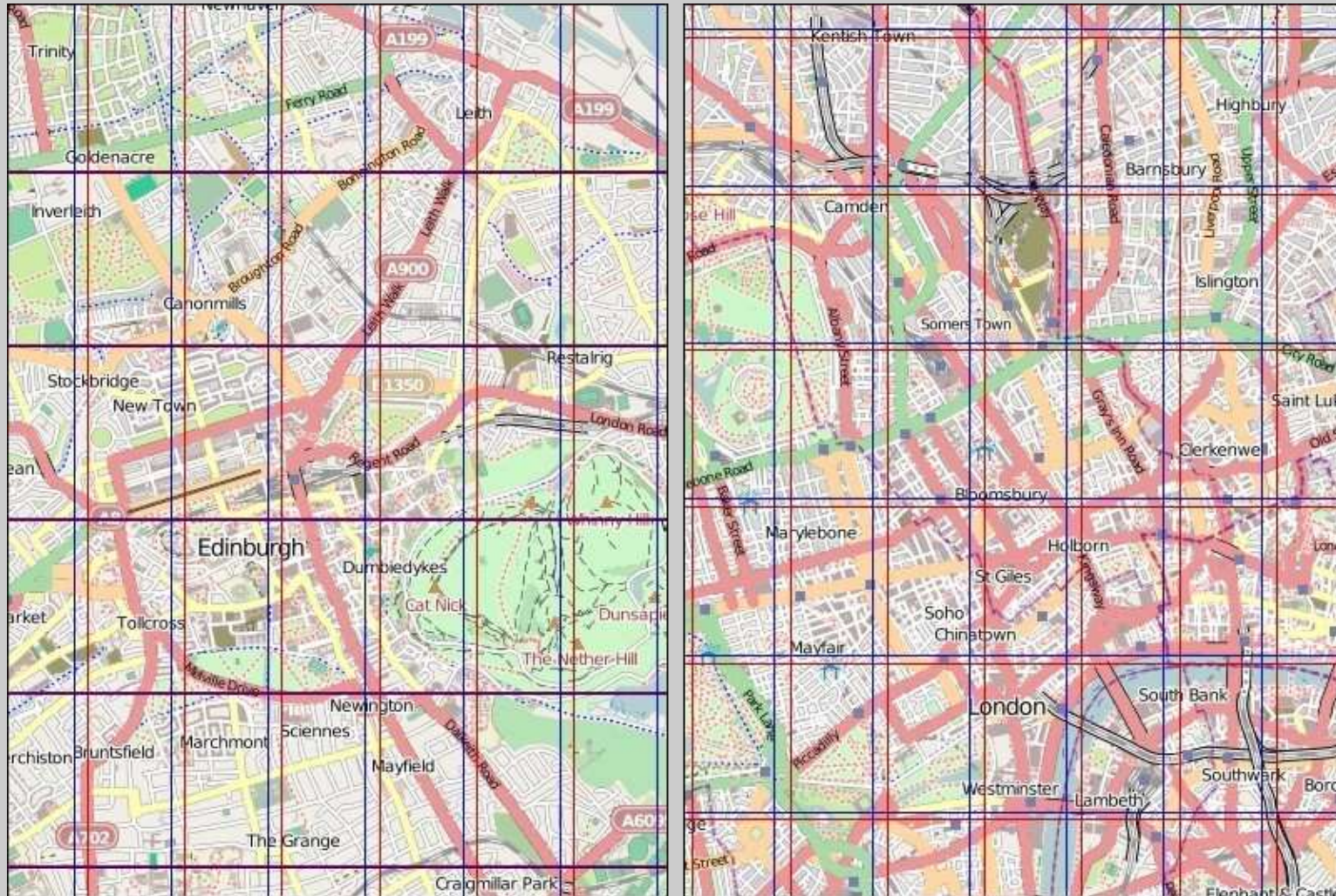
- WGS84: That's what GPS uses
- OSGB36: That's what Ordnance Survey uses
- ED50: That's what we use in most of Europe

# Greenwich Meridian



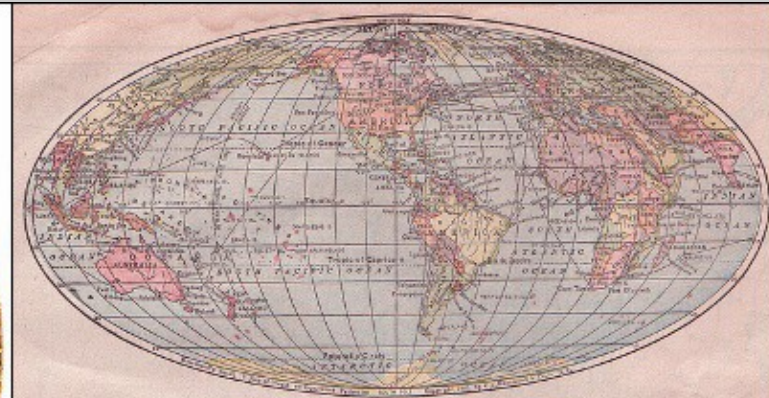
Greenwich Meridian  
IRTS Meridian

# Geoids and Coordinates



Different geoids give different coordinates for places

# Map Projections



MAP OF THE WORLD ON THE EQUIVALENT PROJECTION

#### EXPLANATORY NOTE

This new map presents a new and more correct view of the geographical area of the earth, not unlike that which would be displayed by the planisphere of right ascension and declination.

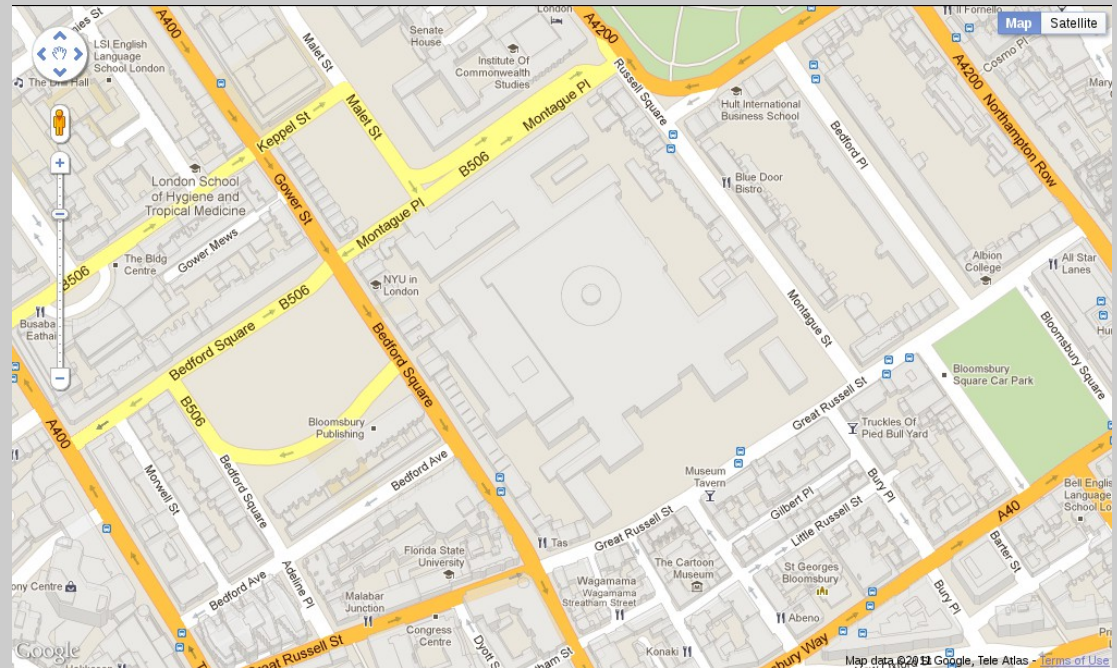
On this map all areas appear in their true proportion. For example, Greenland, which, on Mercator's Projection, seems to be larger than South America, here is shown in its relative proportion of size to that continent, being less than one-sixth of the latter's area. In fact, the area of the world is here shown in its true proportion.



Different projections have different strengths

# Showing a Map

## Google Maps



```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
<style type="text/css">
  html { height: 100% }
  body { height: 100%; margin: 0px; padding: 0px }
  #map_canvas { height: 100% }
</style>
<script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=false">
</script>
<script type="text/javascript">
  function initialize() {
    var latlng = new google.maps.LatLng(51.51922, -0.12736);
    var myOptions = {
      zoom: 17, center: latlng,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    var map = new google.maps.Map(document.getElementById("map_canvas"), myOptions);
  }
</script>
</head>
<body onload="initialize()">
  <div id="map_canvas" style="width:100%; height:100%"></div>
</body>
</html>
```

# Showing a Map

## OpenLayers

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="EN">
  <head>
    <style>
      html,body { margin: 0; padding: 0; width: 1004px; height: 590px; }
      #map { width: 100%; height: 100%; border: 1px solid black; float: left; z-index: -1; }
      div.olControlAttribution { bottom: 0.5em; font-size: 70%; }
    </style>
    <script src='OpenLayers.js'></script>
    <script src='osm/OpenStreetMap.js'></script>
    <script type="text/javascript">
      var map; //complex object of type OpenLayers.Map

      var lat=51.51922
      var lon=-0.12736
      var zoom=17

      function init() {
        map = new OpenLayers.Map ("map", {
          controls:[
            new OpenLayers.Control.PanZoomBar(),
            new OpenLayers.Control.Attribution()],
          projection: new OpenLayers.Projection("EPSG:900913"),
          displayProjection: new OpenLayers.Projection("EPSG:4326")
        } );

        layerMapnik = new OpenLayers.Layer.OSM.Mapnik("Mapnik");
        map.addLayer(layerMapnik);

        var lonLat = new OpenLayers.LonLat(lon, lat).
          transform(map.displayProjection, map.projection);
        map.setCenter(lonLat, zoom);
      }
    </script>
  </head>
  <body onload="init();">
    <div id='map'></div>
```



# Showing a Map

## Leaflet

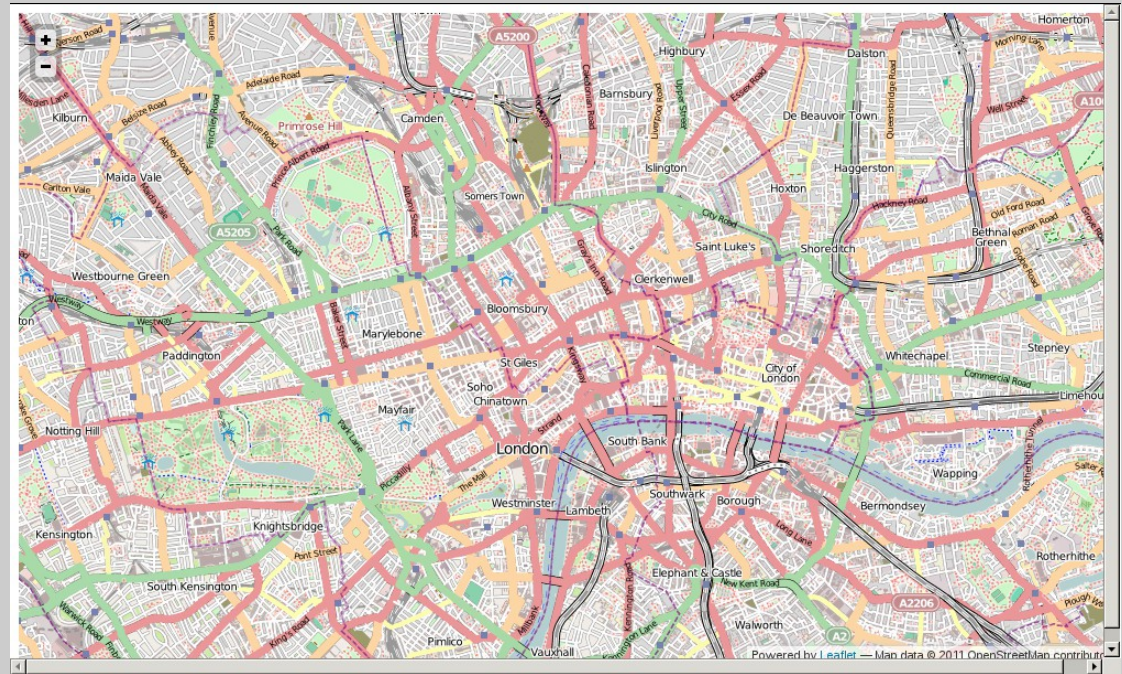
```
<!DOCTYPE html>
<html>
<head>
  <title>Leaflet Quick Start Guide Example</title>
  <meta charset="utf-8" />

  <link rel="stylesheet" href="leaflet/leaflet.css" />
  <!--[if lte IE 8]><link rel="stylesheet" href="leaflet/leaflet.ie.css" /><![endif]-->
  <script src="leaflet/leaflet.js"></script>
</head>
<body>
  <div id="map" style="width: 1004px; height: 590px"></div>
  <script type="text/javascript">
    var map = new L.Map('map');

    var osmUrl = 'http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
        osmAttrib = 'Map data &copy; 2011 OpenStreetMap contributors',
        osm = new L.TileLayer(osmUrl, {maxZoom: 18, attribution: osmAttrib});

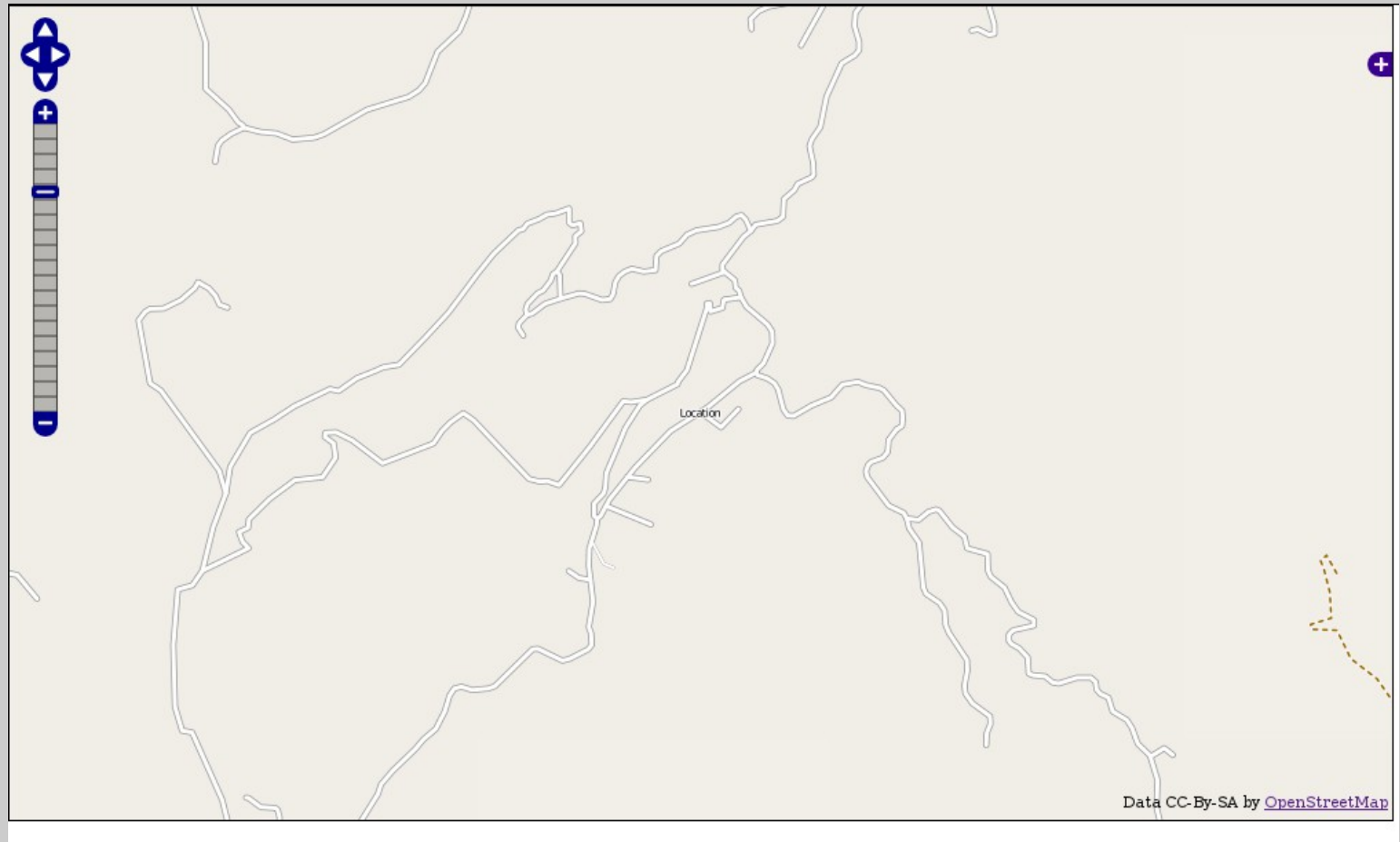
    map.setView(new L.LatLng(51.5179, -0.12), 13).addLayer(osm);

    var popup = new L.Popup();
  </script>
</body>
</html>
```



# Showing a Map

## Looking up latitude and longitude from a location



```
<?php
$name = urlencode( ':-:location:-:' );
$baseUrl = 'http://nominatim.openstreetmap.org/search?format=json&q=';
$data = file_get_contents( "{$baseUrl}{$name}&limit=1" );
$json = json_decode( $data );
$lat = $json[0]->lat;
$lon = $json[0]->lon;
?>
var lat=<?php printf( '%0.3f', $lat ); ?>
var lon=<?php printf( '%0.3f', $lon ); ?>
<?php var_dump( $json[0] ); ?>
```

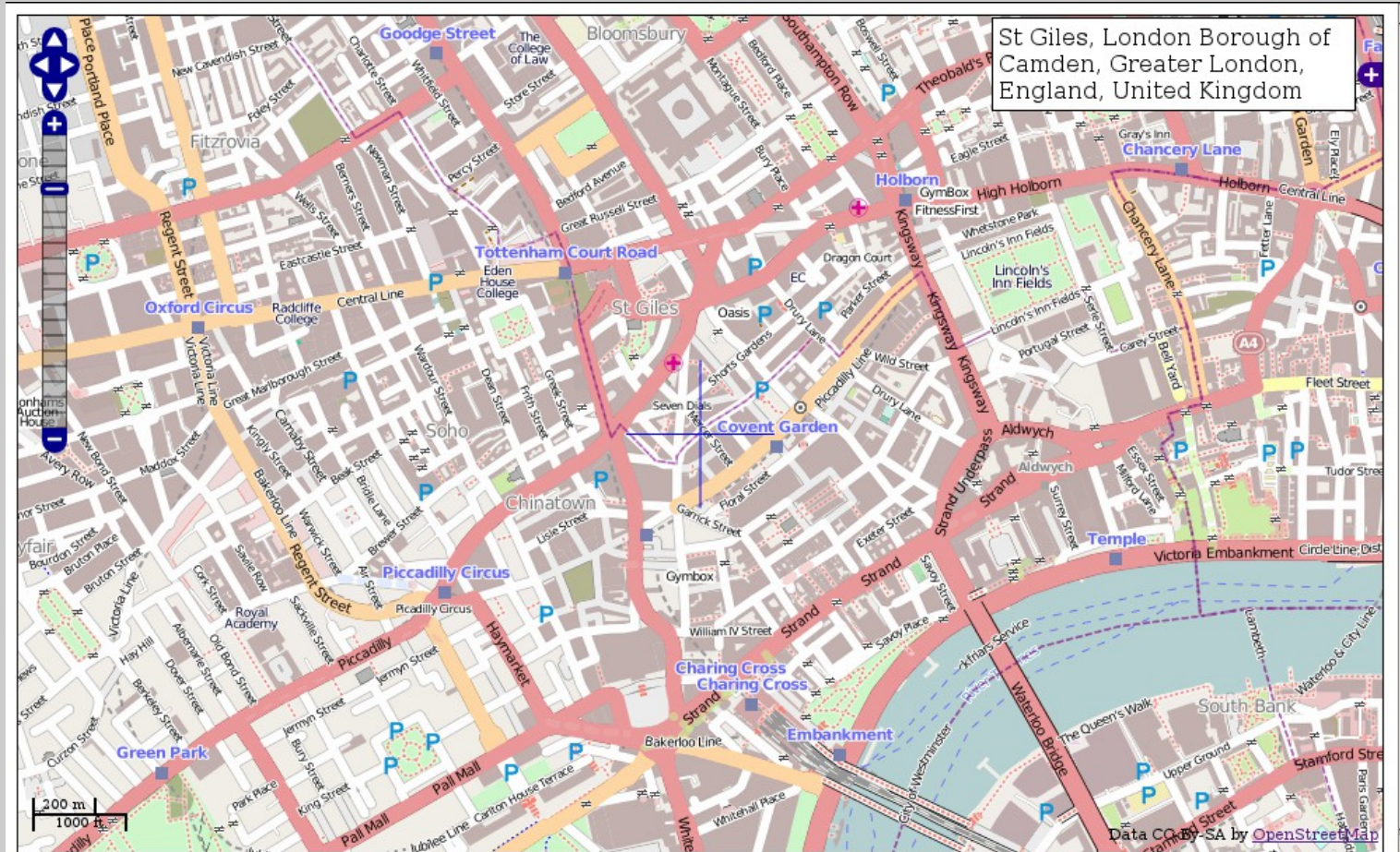
# Looking up latitude/longitude

## Different services

- Nominatim:<http://nominatim.openstreetmap.org/search?format=json&limit=1&q=London>
- Yahoo:[http://where.yahooapis.com/geocode?flags=GJT&appid=\[yourappidhere\]&q=London](http://where.yahooapis.com/geocode?flags=GJT&appid=[yourappidhere]&q=London)

# Reverse Geocoding

## Finding a name for coordinates



# Finding a name for the current location

## Different services

- Geonames:[http://ws.geonames.org/findNearbyPlaceNameJSON?](http://ws.geonames.org/findNearbyPlaceNameJSON?username=derick&style=full&lat={ $lat } &lng={ $lon })  
`username=derick&style=full&lat={ $lat } &lng={ $lon }`
- Nominatim:[http://nominatim.openstreetmap.org/reverse?](http://nominatim.openstreetmap.org/reverse?format=json&lat={ $lat } &lon={ $lon } &zoom={ $z })  
`format=json&lat={ $lat } &lon={ $lon } &zoom={ $z }`
- Yahoo:[http://where.yahooapis.com/geocode?](http://where.yahooapis.com/geocode?gflags=R&flags=GJQT&q={ $lat }, { $lon })  
`gflags=R&flags=GJQT&q={ $lat }, { $lon }`

# Finding the user

## Using JavaScript to locate the user

```
function getPosition()
{
    navigator.geolocation.getCurrentPosition(iKnowWhereYouAre, notTheFaintestClue,
    {timeout:30000});
}

function notTheFaintestClue()
{
}

function iKnowWhereYouAre(position)
{
    var lonLat = new OpenLayers.LonLat(
        position.coords.longitude, position.coords.latitude
    ).transform(map.displayProjection, map.projection);
    map.setCenter(lonLat, zoom);

    center = map.getCenter().
        transform(map.getProjectionObject(), new OpenLayers.Projection("EPSG:4326"));

    factor = Math.cos(center.lat / (180/Math.PI)), 10 + map.getZoom() * 2;

    multiFeature = new OpenLayers.Feature.Vector(
        OpenLayers.Geometry.Polygon.createRegularPolygon(
            new OpenLayers.Geometry.Point(
                center.lon, center.lat
            ).transform(new OpenLayers.Projection("EPSG:4326"), map.getProjectionObject()),
            position.coords.accuracy / factor, 10
        ),
        {
            color: 'blue',
            align: 'rt'
        }
    );

    vectorLayer.removeAllFeatures();
    vectorLayer.drawFeature(multiFeature);
    vectorLayer.addFeatures([multiFeature]);
}
```



# Google Geo-location Service

```
<?php
$request = array(
    'version' => '1.1.0',
    'host' => 'example.com',
    'wifi_towers' => array(
        array( 'ssid' => 'ZyXEL_3934rar', 'mac_address' => "00:02:CF:E4:60:CE" )
    )
);
$c = curl_init();
curl_setopt( $c, CURLOPT_URL, 'https://www.google.com/loc/json' );
curl_setopt( $c, CURLOPT_POST, 1 );
curl_setopt( $c, CURLOPT_POSTFIELDS, json_encode( $request ) );
curl_setopt( $c, CURLOPT_RETURNTRANSFER, true );
var_dump( json_decode( curl_exec( $c ) ) );
```

[http://code.google.com/intl/es-ES/apis/gears/geolocation\\_network\\_protocol.html](http://code.google.com/intl/es-ES/apis/gears/geolocation_network_protocol.html)



- "Wikipedia for Map Data"
- Licensed under the Creative Commons Attribution-ShareAlike 2.0 licence (CC-BY-SA):  
You are free to copy, distribute, transmit and adapt our maps and data, as long as you credit OpenStreetMap and its contributors. If you alter or build upon our maps or data, you may distribute the result only under the same licence.
- Rendered map:
- A lot of data is not rendered, but is available.

# Fetching OSM data

```
wget
http://open.mapquestapi.com/xapi/api/0.6/node
[amenity=pub]
[bbox=-2.401,53.394,-2.104,53.551]
-O pubs.osm
<?xml version='1.0' encoding='UTF-8'?>
<osm version='0.6' generator='xapi: OSM Extended API 2.0' attribution='http://wiki.openstreetmap.org/wiki/Attribution'
xmlns:xapi='http://www.informationfreeway.org/xapi/0.6' xapi:uri='/api/0.6/node[amenity=pub][bbox=-2.401,53.394,-
2.104,53.551]' xapi:planetDate='20101006' xapi:copyright='2010 OpenStreetMap contributors' xapi:license='Creative commons
CC-BY-SA 2.0' xapi:bugs='For assistance or to report bugs contact 80n80n@gmail.com' xapi:instance='zappyHyper'>
<bounds minlat='53.394' minlon='-2.401' maxlat='53.551' maxlon='-2.104'/>
  <node id='275332052' lat='53.548238' lon='-2.3958373' version='2' changeset='4395635'
    user='Steeley' uid='101150' visible='true' timestamp='2010-04-11T17:08:16Z'>
    <tag k='amenity' v='pub'/>
    <tag k='name' v='The Saddle'/>
  </node>
  ...
  <node id='30732192' lat='53.4647746' lon='-2.2319186' version='3' changeset='5810586'
    user='geordiemanc' uid='345640' visible='true' timestamp='2010-09-18T11:12:50Z'>
    <tag k='address' v='325 Oxford Road'/>
    <tag k='amenity' v='pub'/>
    <tag k='name' v='Kro Bar'/>
    <tag k='phone' v='01612743100'/>
    <tag k='postal_code' v='M13 9PG'/>
    <tag k='real_ale' v='yes'/>
  </node>
```

# The Data

- Nodes (Lat/Lon point)  
`<node id='459517295' lat='50.0100766' lon='8.3162402'  
user='WoGo' timestamp='2009-08-09T11:45:33Z'  
uid='152395' version='1' changeset='2083951'>`
- Ways (Ordered interconnection of nodes)
- Areas (Closed ways)  
`<way id='76174399' user='Derick Rethans' uid='37137'  
timestamp='2010-09-06T08:30:14Z' version='1'  
changeset='5695697'> <nd ref='898861293'/> <nd  
ref='898861305'/> <nd ref='898861298'/> <nd  
ref='898861315'/> <nd ref='898861293'/> ... </way>`
- Tags (Describe an element)  
`<tag k='addr:housenumber' v='375'/> <tag  
k='addr:street' v='Kilburn High Road'/> <tag k='amenity'  
v='pub'/> <tag k='building' v='yes'/> <tag k='name'  
v='North London Tavern'/>`

# Massage the Data

## Process:

- Use XAPI to fetch data
- Parse XML file with PHP into a DB
- Query database
- Show data
- Profit!

# Finding Food

```
function init() {
    map = new OpenLayers.Map ("map", {
        eventListeners: {
            "moveend": moveEndEvent
        },
        controls:[
function changeQuery()
{
    cuisine = document.getElementById('amenity').value;
    radiusInput = document.getElementById('radius');
    source = document.getElementById('source').value;

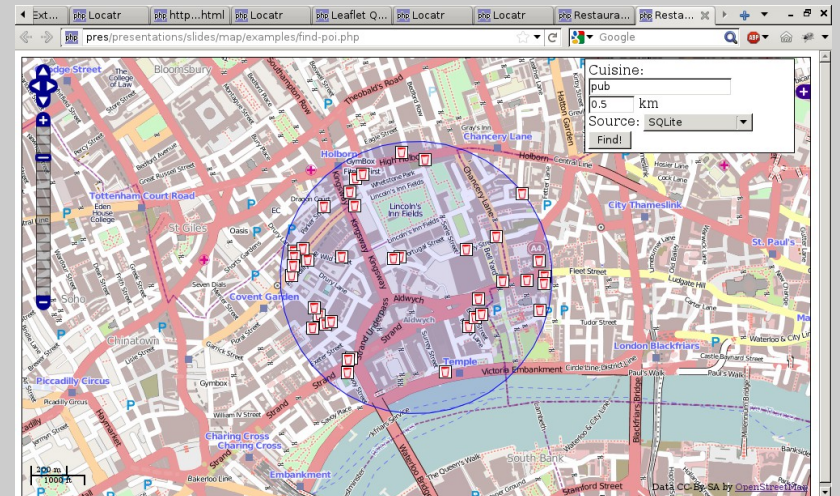
    if (source == 'sqlite') { script = 'fetch.php'; }
    if (source == 'mysql') { script = 'fetch-mysql.php'; }
    if (source == 'mongo') { script = 'fetch-mongo.php'; }
    if (source == 'mongo2') { script = 'fetch-mongo-fixed.php'; }

    center = map.getCenter().transform(map.getProjectionObject(), new OpenLayers.Projection("EPSG:4326"));
    pois.destroy();
    pois = new OpenLayers.Layer.Text( "The Shops", {
        location: "." + script + "?cuisine=" + cuisine +
            '&lat=' + center.lat + '&lon=' + center.lon + '&d=' + radiusInput.value,
        projection: map.displayProjection
    });
    map.addLayer(pois);

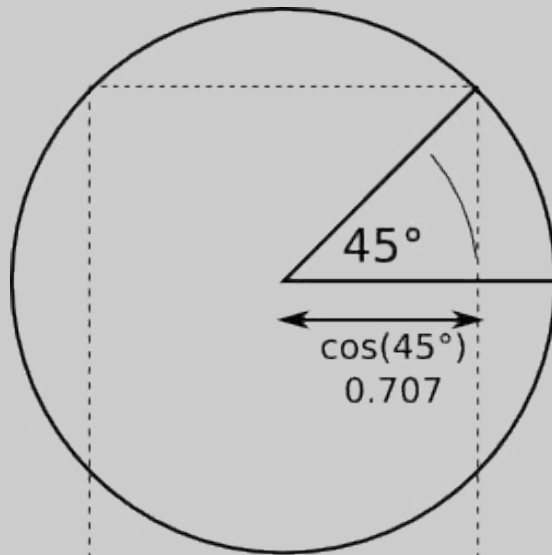
    multiFeature = new OpenLayers.Feature.Vector(
        OpenLayers.Geometry.Polygon.createRegularPolygon(
            new OpenLayers.Geometry.Point(center.lon,center.lat).transform(new OpenLayers.Projection("EPSG:4326"),
map.getProjectionObject()),
            radiusInput.value * 1000 / Math.cos(center.lat / (180/Math.PI)), 10 + map.getZoom() * 2, 10
        ),
        {
            color: 'blue',
            align: 'rt'
        }
    );

    vectorLayer.removeAllFeatures();
    vectorLayer.drawFeature(multiFeature);
    vectorLayer.addFeatures([multiFeature]);
}

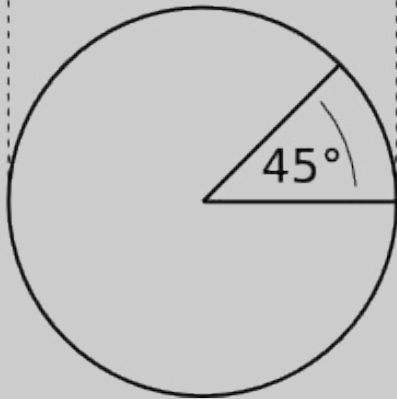
function moveEndEvent(event)
{
    changeQuery();
}
```



# Distances are tricky



$$\frac{1}{8} \cdot 2\pi \cdot 6371\text{km} = 5003\text{km}$$



$$\frac{1}{8} \cdot 2\pi \cdot 6371\text{km} \cdot \cos(45^\circ) = 3538\text{km}$$

note: km / miles  $\approx \cos(51.5)$

# Finding Food

## Getting The Data: SQLite

```
<?php
include 'distance.php';
header('Content-type: text/plain');
require '/home/derick/dev/zetacomponents/trunk/Base/src/ezc_bootstrap.php';
$d = ezcdbFactory::create( 'sqlite://' . dirname( __FILE__ ) . '/pois.sqlite' );

$wantedD = isset($_GET['d']) ? $_GET['d']: 1;
$q = $d->createSelectQuery();
$q->select('*')->from('poi');
if ( $_GET['cuisine'] !== 'all' )
{
    $q->where($q->expr->eq('cuisine', $q->bindValue( $_GET['cuisine'] ) ) );
}
$s = $q->prepare();
$s->execute();
echo "lat\tlon\ttitle\tdescription\ticonSize\ticonOffset\ticon\r\n";
foreach( $s as $res ) {
    $e = distance2($_GET['lat'], $_GET['lon'], $res['lat'], $res['lon'] );
    if ( $e < $wantedD ) {
        echo $res['lat'], "\t", $res['lon'], "\t", $res['name'], "\t", sprintf('%.2f', $e). " km away\t16,16\t-8,-8\tpub.png\r\n";
    }
}
```

## Calculating Distance

```
<?php
function distance2($latA, $lonA, $latB, $lonB)
{
    $latA = deg2rad($latA);
    $lonA = deg2rad($lonA);
    $latB = deg2rad($latB);
    $lonB = deg2rad($lonB);

    $deltaLat = ($latA - $latB);
    $deltaLon = ($lonA - $lonB);

    $d = sin($deltaLat/2) * sin($deltaLat/2) +
        cos($latA) * cos($latB) * sin($deltaLon/2) * sin($deltaLon/2);
    $d = 2 * asin(sqrt($d));
    return $d * 6371.01;
}
```

# Finding Food

## Getting The Data: MySQL

```
<?php
include 'distance.php';
header('Content-type: text/plain');
require '/home/derick/dev/zetacomponents/trunk/Base/src/ezc_bootstrap.php';
$d = ezcdbFactory::create( 'mysql://root:root@localhost/geolocation' );

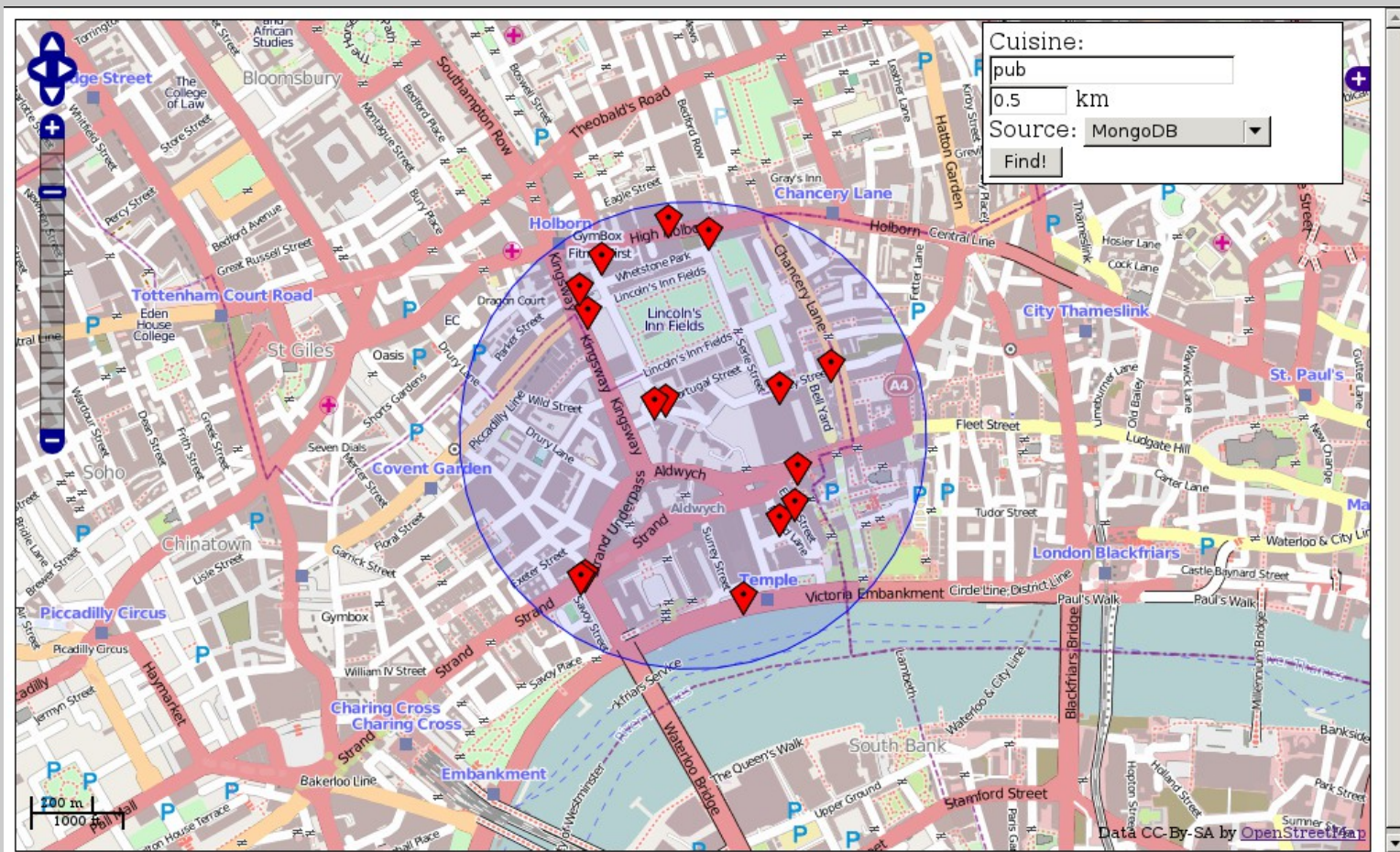
$wantedD = isset($_GET['d']) ? $_GET['d']: 1;
$q = $d->createSelectQuery();
$q->select('*', "DISTANCE({$_GET['lat']},{$_GET['lon']}, lat, lon) as dist")->from('poi');
if ( $_GET['cuisine'] !== 'all' )
{
    $q->where($q->expr->eq('cuisine', $q->bindValue( $_GET['cuisine'] ) ) );
}
$s = $q->prepare();
$s->execute();
```

## Stored Procedure

```
delimiter //

CREATE FUNCTION distance (latA double, lonA double, latB double, LonB double)
    RETURNS double DETERMINISTIC
BEGIN
    SET @RlatA = radians(latA);
    SET @RlonA = radians(lonA);
    SET @RlatB = radians(latB);
    SET @RlonB = radians(LonB);
    SET @deltaLat = @RlatA - @RlatB;
    SET @deltaLon = @RlonA - @RlonB;
    SET @d = SIN(@deltaLat/2) * SIN(@deltaLat/2) +
        COS(@RlatA) * COS(@RlatB) * SIN(@deltaLon/2)*SIN(@deltaLon/2);
    RETURN 2 * ASIN(SQRT(@d)) * 6371.01;
END//
```

# Finding Food



# Finding Food

## Getting The Data: MongoDB

```
<?php
header('Content-type: text/plain');
$m = new MongoClient( 'mongodb://localhost:27017' );
$d = $m->selectDb( 'geolocation' );

$wantedD = isset($_GET['d']) ? $_GET['d']: 1;

$query = array( 'cuisine' => $_GET['cuisine'] );
if ( $_GET['cuisine'] == 'all' )
{
    $query = array();
}

$s = $d->command(
    array(
        'geoNear' => 'poi',
        'near' => array( $_GET['lat'], $_GET['lon'] ),
        'num' => 10000,
        'maxDistance' => $wantedD * (360 / (2*M_PI*6371.01)), // km to °
        'query' => $query,
    )
);
echo "lat\tlon\ttitle\tdescription\r\n";
foreach( $s['results'] as $res ) {
    if (isset($res['obj']['name']) ) {
        echo $res['obj']['loc'][0], "\t", $res['obj']['loc'][1], "\t", $res['obj']['name'], "\t", sprintf('real: %.4f mongo: %.4f', $e, $res['dis'] / (360 / (2*M_PI*6371.01)),
    }
}
```

## Spatial Index

```
db.poi.ensureIndex( { poi : '2d' } );
```

```
$s = $d->command(
    array(
        'geoNear' => 'poi',
        'near' => array( $_GET['lat'], $_GET['lon'] ),
        'num' => 10000,
        'maxDistance' => $wantedD * (360 / (2*M_PI*6371.01)), // km to °
        'query' => $query,
    )
);
```

## Geospatial Index (since 1.7)

```
function newImageMarker(url, lat, lon)
{
    w = 85 - ((19-map.getZoom())*4);
    size = new OpenLayers.Size(w,w);
    offset = new OpenLayers.Pixel(-(size.w/2), -(size.h/2));
    icon = new OpenLayers.Icon(url, size, offset);

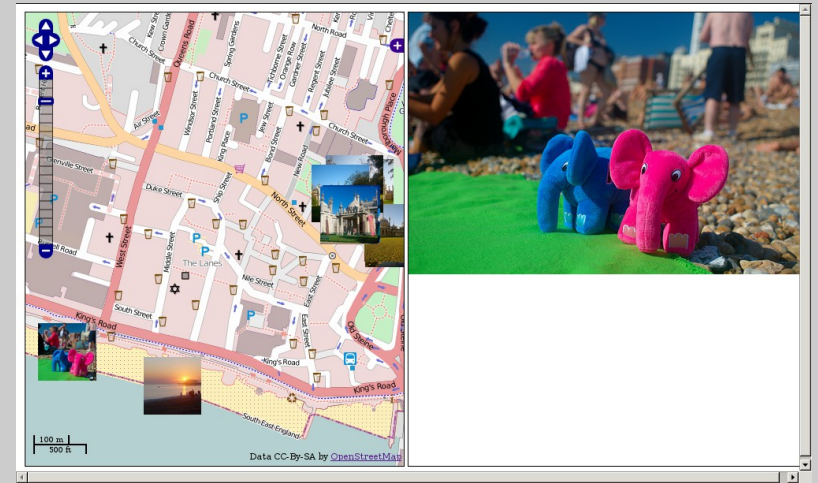
    marker = new OpenLayers.Marker(
        new OpenLayers.LonLat(lon, lat)
        .transform(
            new OpenLayers.Projection("EPSG:4326"),
            map.getProjectionObject()
        ),
        icon.clone()
    );
    marker.events.register(
        'mousedown',
        marker,
        function(evt) { showImage(this.icon); OpenLayers.Event.stop(evt); }
    );
    markers.addMarker(marker);
}

function changeQuery()
{
    markers.clearMarkers();
    $.getJSON('fetch-flickr.php', function(data) {
        $.each(data.items, function(i,item){
            newImageMarker(item.url, item.lat, item.lon);
        });
    });
}

<?php
$d = ezcdbFactory::create( 'sqlite://' . dirname( __FILE__ ) . '/presentations/slides/map/examples/photos.sqlite' );

$q = $d->createSelectQuery();
$q->select('*')->from('photo')->orderBy( 'date_taken', ezcdbQuerySelect::DESC )->limit(100);
$s = $q->prepare();
$s->execute();

$items = array();
foreach ( $s as $photo )
{
    $items[] = array(
        'lon' => $photo['lon'],
        'lat' => $photo['lat'],
        'url' => $photo['thumb_url']
    );
}
echo json_encode(array( 'items' => $items ) );
```



derick@derickrethans.nl - twitter: @derickr

<http://derickrethans.nl/talks.html>

<http://joind.in/3841>

- <http://openstreetmap.org>
- <http://mapref.org>
- <http://dev.openlayers.org/docs/files/OpenLayers-js.html>
- <http://data.london.gov.uk/taxonomy/categories/transport>
- <http://www.flickr.com/services/api/>
- <http://www.ordnancesurvey.co.uk/oswebsite/gps/information/coordinatesystemsinfo/guidecontents/index.html>
- [http://en.wikipedia.org/wiki/Helmert\\_transformation](http://en.wikipedia.org/wiki/Helmert_transformation)
- [http://wiki.openstreetmap.org/wiki/OSTN02\\_for\\_PHP](http://wiki.openstreetmap.org/wiki/OSTN02_for_PHP)
- <http://leaflet.cloudmade.com/examples/quick-start.html>
- <http://code.google.com/apis/maps/documentation/javascript/>
- <http://wiki.openstreetmap.org/wiki/Nominatim>
- <http://developer.yahoo.com/geo/placementfinder/guide/>
- <http://www.geonames.org/export/web-services.html>
- [http://code.google.com/intl/es-ES/apis/gears/geolocation\\_network\\_protocol.html](http://code.google.com/intl/es-ES/apis/gears/geolocation_network_protocol.html)
- <http://www.mongodb.org/display/DOCS/Geospatial+Indexing>
- <http://en.wikipedia.org/wiki/Gpx>

# Getting the WLAN info

```
<?php
define( 'NM', "org.freedesktop.NetworkManager" );
$d = new Dbus( Dbus::BUS_SYSTEM, true );
$n = $d->createProxy( NM, "/org/freedesktop/NetworkManager", NM);
$wifi = array();
foreach ( $n->GetDevices()->getData() as $device )
{
    $device = $device->getData();
    $dev = $d->createProxy( NM, $device, "org.freedesktop.DBus.Properties");
    $type = $dev->Get( NM . ".Device", "DeviceType" )->getData();
    if ( $type == 2 ) // WI-FI
    {
        $wifiDev = $d->createProxy( NM, $device, NM . ".Device.Wireless");
        foreach( $wifiDev->GetAccessPoints()->getData() as $ap )
        {
            $apDev = $d->createProxy( NM, $ap->getData(), "org.freedesktop.DBus.Properties");
            $props = $apDev->GetAll( NM . ".AccessPoint" )->getData();
            $ssid = '';
            foreach( $props['Ssid']->getData()->getData() as $n )
            {
                $ssid .= chr($n);
            }
            $wifi[] = array('ssid' => $ssid, "mac_address" => $props['HwAddress']->getData() );
        }
    }
}

$request = array( 'version' => '1.1.0', 'host' => 'example.com', 'wifi_towers' => $wifi );

$c = curl_init();
curl_setopt( $c, CURLOPT_URL, 'https://www.google.com/loc/json' );
curl_setopt( $c, CURLOPT_POST, 1 );
curl_setopt( $c, CURLOPT_POSTFIELDS, json_encode( $request ) );
curl_setopt( $c, CURLOPT_RETURNTRANSFER, true );
$result = json_decode( curl_exec( $c ) )->location;
echo "<a href='http://openstreetmap.org/?lat={$result->latitude}&lon={$result->longitude}&zoom=18'>here</a>\n";
?>
```