SAVING GAIA

w/jQuery Mobile, OpenLayers and GeoDjango

CALVIN CHENG

www.calvinx.com

www.od-eon.com/blogs/calvin

1 Context

2 Technology Stack

3 Installation

4 Django GeoModel & ORM

5 Geospatial Libraries

6 Utilities

7 GeoAdmin

8 OpenLayers

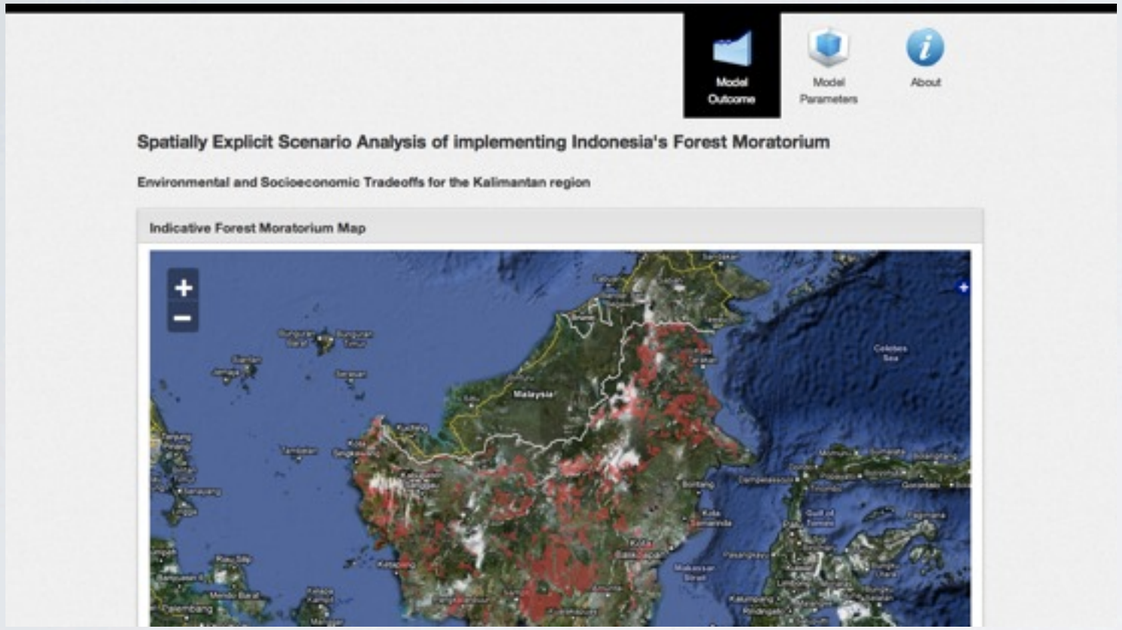
9 jQuery (Mobile)

10 Possibilities

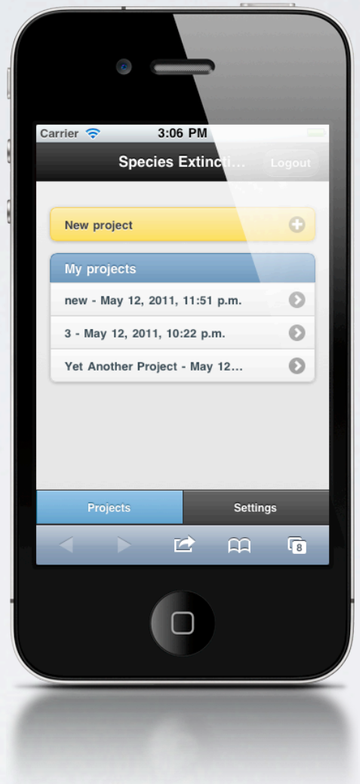
11 Conclusion

1 Context

REDDCalculator.com (Beta)



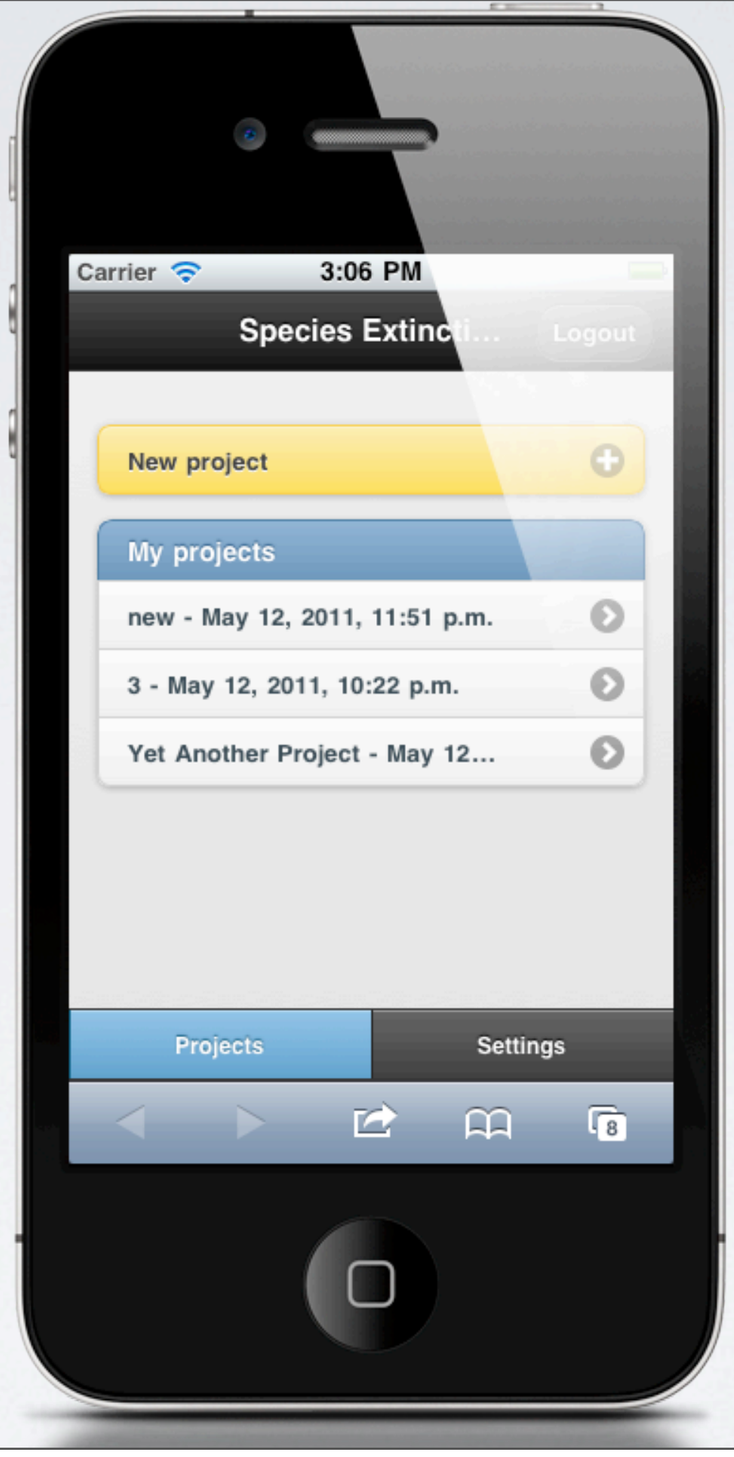
SEC (Closed Beta)



LUC (Coming Soon!)

ENVIRONMENTAL iTOOLS

- Spatially Explicit Scenario Analysis: Norway-Indonesia's Forest Moratorium
- Species Extinction Calculator (SEC)
- Land Use Calculator (LUC)


Carrier
3:06 PM
Species Extincti...
Logout
New project
My projects
new - May 12, 2011, 11:51 p.m.
3 - May 12, 2011, 10:22 p.m.
Yet Another Project - May 12...
Projects
Settings

SPECIES WHA--??

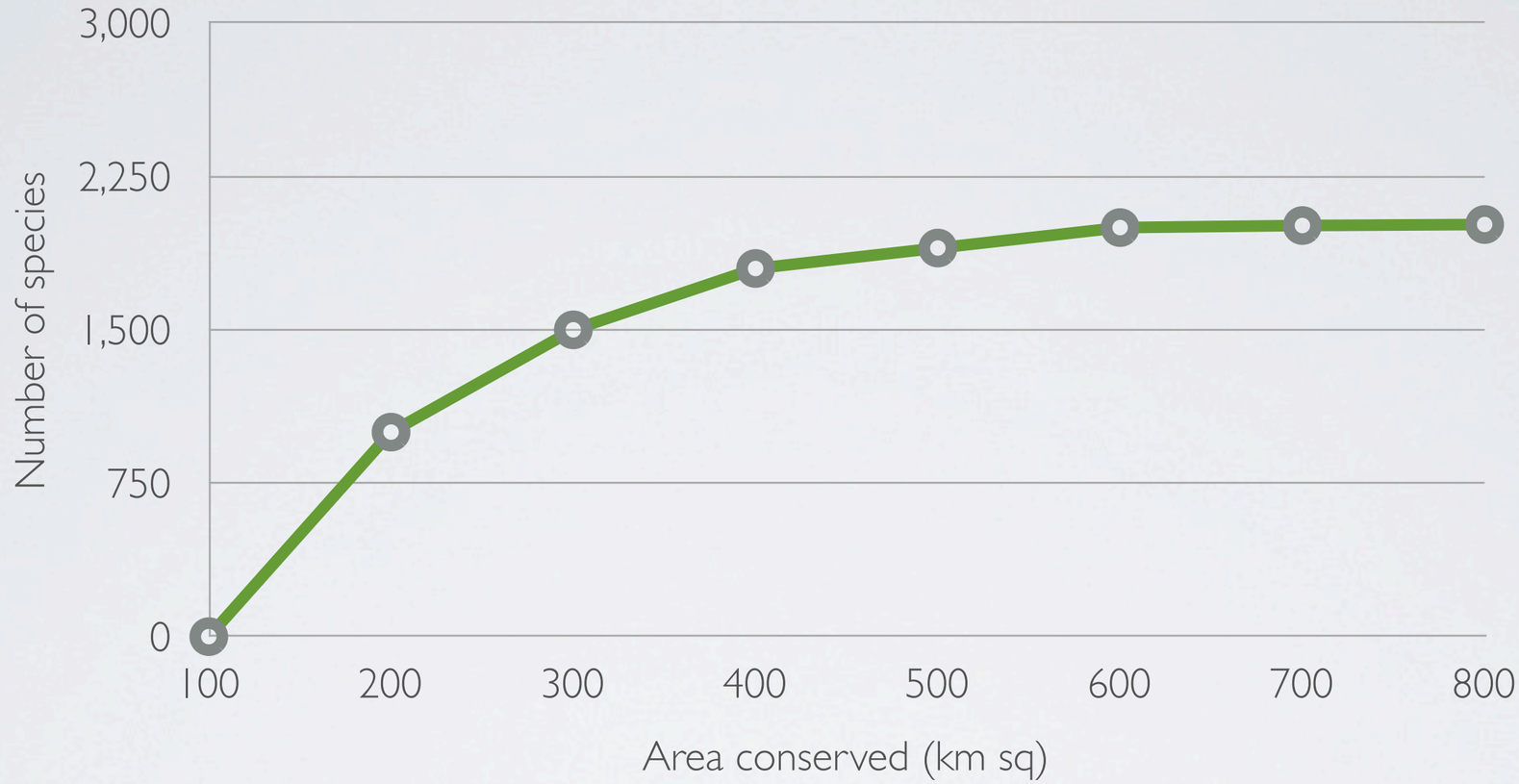
Species Extinction Calculator (SEC)

z

Species $S=cA^z$

- Arrhenius (1920)

Area



SPECIES-AREA

More habitats conserved (Area) = More fauna diversity (Species)

Thursday, June 23, 2011

$$\frac{S_{new}}{S_{original}} = \left(\frac{A_{new}}{A_{original}} \right)^{z}$$

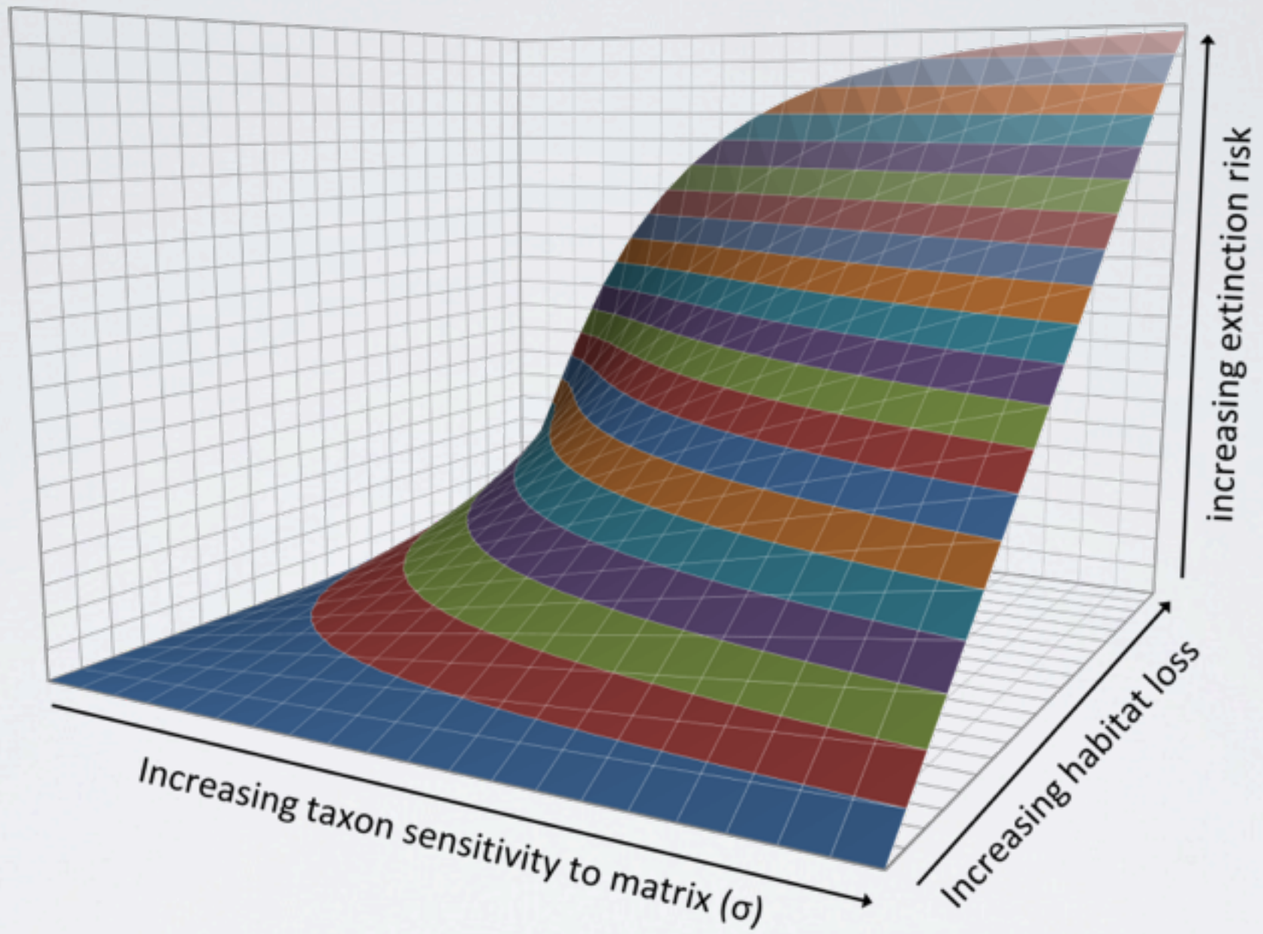
Problem #1: Matrix effects

Problem #2: Edge effects

Taxon-specific responses to landscape matrix

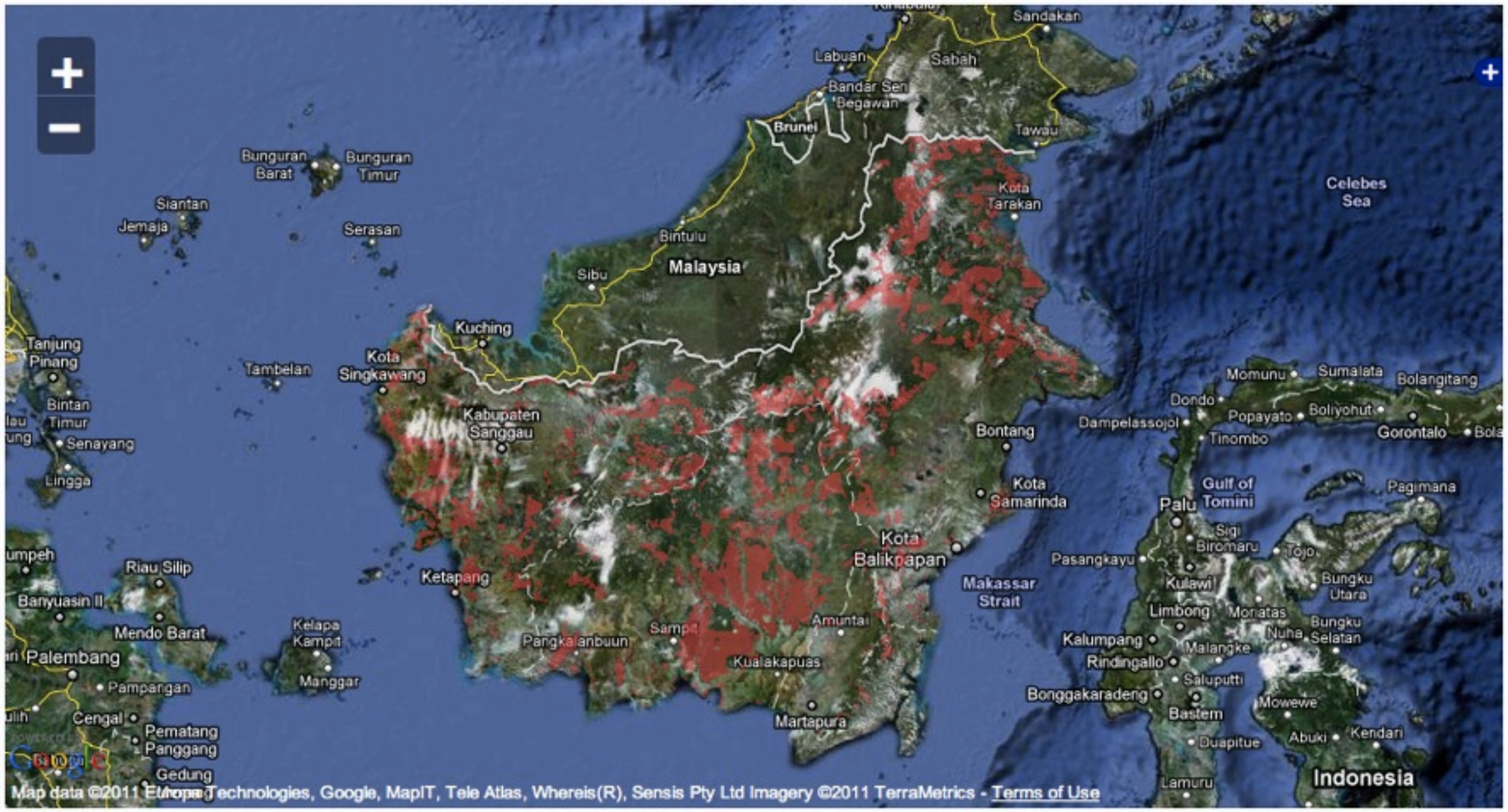
Negative effects of forest edges

$$\frac{S_{new}}{S_{original}} = \left(\frac{A_{new} - \delta \cdot \sum_{j=1}^{M} \beta_j}{A_{original}} \right)^{\gamma \cdot \sum_{i=1}^{N} \rho_i \cdot \sigma_i}$$



MATRIX-CALIBRATED SPECIES-AREA

Accounts for taxon-specific responses to different components of the landscape



EXAMPLE OF AN AREA?

Conservation Forests, National Reserves etc
Borneo 743,330 square kilometres

USERS AND USE-CASE

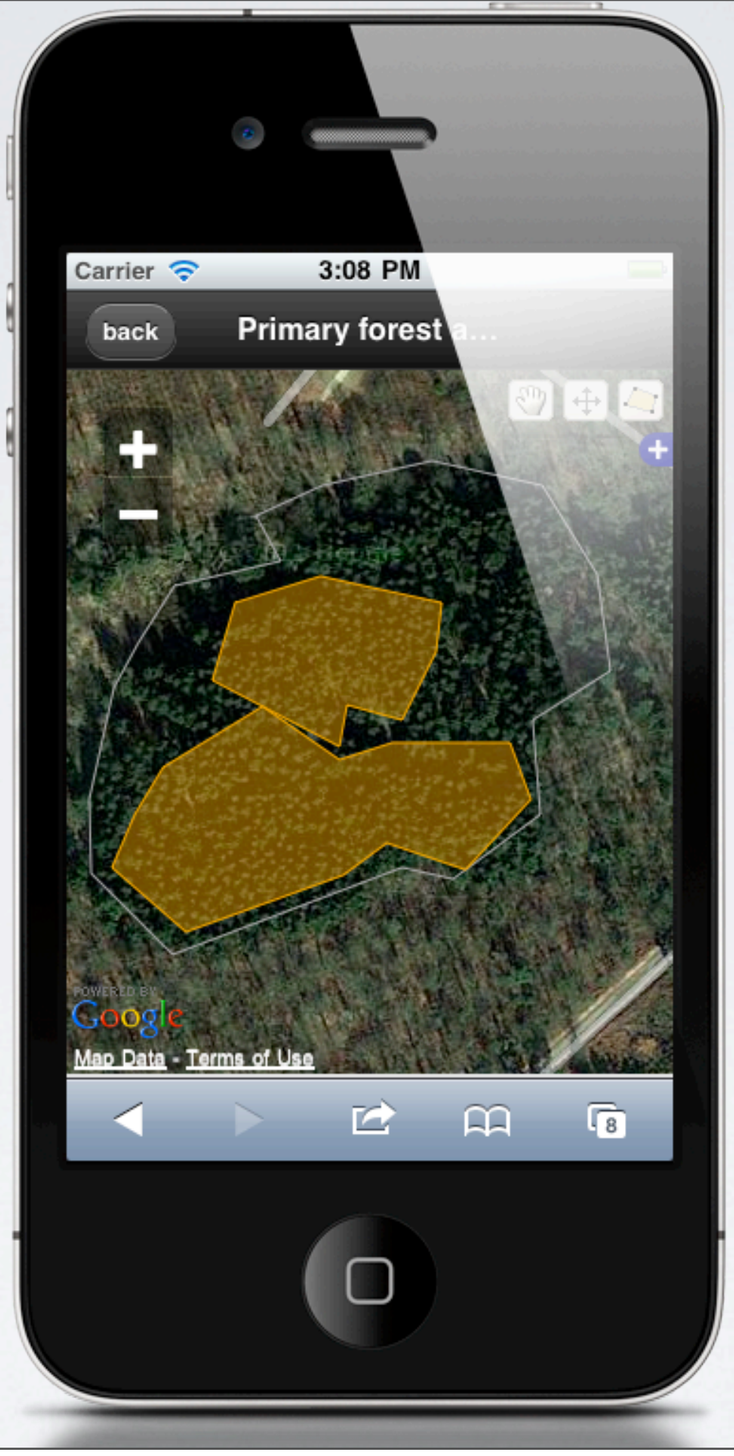
Conservation scientists and **Land owners** define areas which are conserved by plotting polygons on a map source (e.g. Google Maps).

Areas which are slated for development are also marked.

By **quantifying** area conservation goals (areas conserved before and after development) and its impact on the loss of species, SEC is a tool which helps people make better trade-off decisions.

USERS AND USE-CASE

REDDCalculator.com computes the socio-economic trade-offs and results as a consequence of the Norway-Indonesia REDD+ Moratorium.


Carrier
3:08 PM
back
Primary forest a…
+
−
POWERED BY
Google
Map Data - Terms of Use
8

Technology Stack

jQuery & OpenLayers

Layers & Features ('Vector Objects')

Map Source

User Interface

KML | GML | WKT | GeoJSON | GeoRSS | Others

Data (serialized)

GeoDjango

Python Logic

GDAL | OSGeo | mapnik | PROJ.4 | PostgreSQL | PostGIS

C/C++ Libraries
w/ Python Bindings
& Datastore

3

Installation


python™ (2.7)
MacPorts
gentoo linux

(trunk)

PostgreSQL

MacPorts

- sudo -u postgres psql -d yourdatabase_name -f { ... }/**postgis.sql**
- sudo -u postgres psql -d yourdatabase_name -f { ... }/**postgis_comments.sql**
- sudo -u postgres psql -d yourdatabase_name -f { ... }/**spatial_ref_sys.sql**

MacPorts

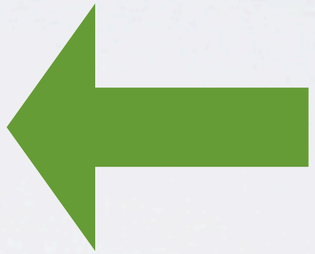
- sudo -u postgres psql -d yourdatabase_name -f /opt/local/var/macports/software/postgis/1.5.2_1+postgresql90/opt/local/share/postgresql90/contrib/postgis-1.5/**postgis.sql**

- sudo -u postgres psql -d yourdatabase_name -f /opt/local/var/macports/software/postgis/1.5.2_1+postgresql90/opt/local/share/postgresql90/contrib/postgis-1.5/**postgis_comments.sql**

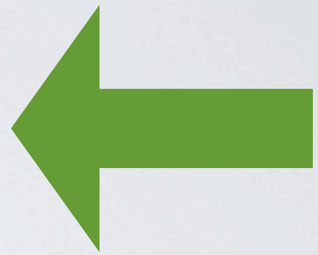
- sudo -u postgres psql -d yourdatabase_name -f /opt/local/var/macports/software/postgis/1.5.2_1+postgresql90/opt/local/share/postgresql90/contrib/postgis-1.5/**spatial_ref_sys.sql**

- Portage exists
- But here's a magical way - https://github.com/stefantalpalaru/dotfiles/tree/master/Gentoo/custom%20overlay/portage/dev-db/postgis - custom postgis ebuild modified by Stefan
- add ebuild to local 'overlay' (a gentoo terminology)
- emerge postgis
- edit configuration file in /etc/conf.d/postgis and add in the postgresql database that you need 'postgis-enabled'
- emerge --config postgis

```
# settings.py

INSTALLED_APPS = (
    # Core Django Apps
    'django.contrib.contenttypes',
    'django.contrib.auth',
    'django.contrib.sessions',
    'django.contrib.sites',
    'django.contrib.admin',
    'django.contrib.admindocs',
    'django.contrib.flatpages',
    'django.contrib.humanize',
    'django.contrib.comments',
    'django.contrib.markup',
    'django.contrib.webdesign',
    'django.contrib.sitemaps',
    'django.contrib.gis',

    # Custom Apps for REDDCalculator.com
    'updatemodel',
    'location',
    'areadata',

    # Generic and reusable apps for most projects
    'south',
    'django_cherrypy_odeon',
    'misc',
    'cron',
    'notify',
    'log',
    'homepage',
    'userprofile',
)
```




```
# settings.py

DATABASES = {
        'default': {
            'ENGINE':
                'django.contrib.gis.db.backends.postgis',
            ‘NAME’: ‘yourdatabase_name’,
            'USER': 'yourdatabase_user’,
            'PASSWORD': 'yourdatabase_password',
            'HOST': 'your_ip_or_domain',
    }
}
```



Django GeoModel & APIs

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

from django.contrib.gis.db import models

```
class Province(models.Model):
    name = models.CharField(max_length=256)
    name_long = models.CharField(max_length=256, null=True)
    island = models.ForeignKey(Location, null=True)
    geom = models.MultiPolygonField(srid=4326, null=True)
    objects = models.GeoManager()
```

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

class Province(models.Model):
    name = models.CharField(max_length=256)
    name_long = models.CharField(max_length=256, null=True)
    island = models.ForeignKey(Location, null=True)
    geom = models.MultiPolygonField(srid=4326, null=True, \
                geography=True)
    objects = models.GeoManager()
```

What are the geometry fields available to us in django?

GEOS: OGC simple feature implementation

- PointField
- MultiPointField

- LineStringField
- MultiLineStringField

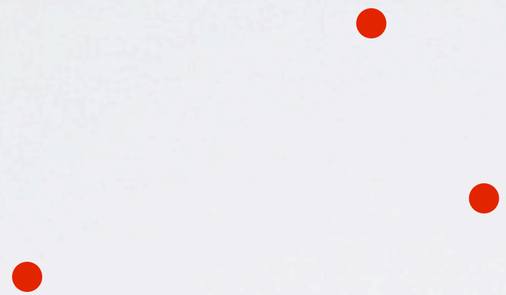
- PolygonField
- MultiPolygonField

- GeometryCollectionField

PointField

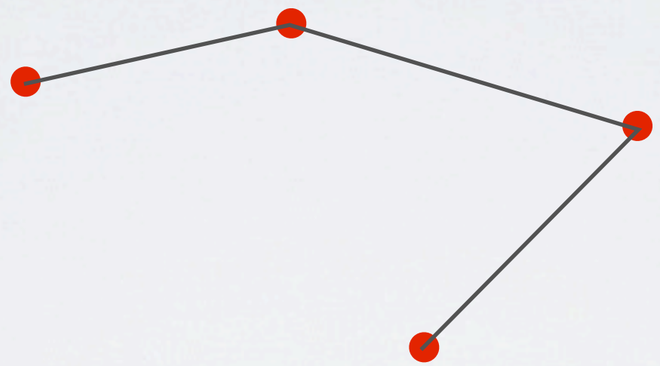
- (x, y)

MultiPointField



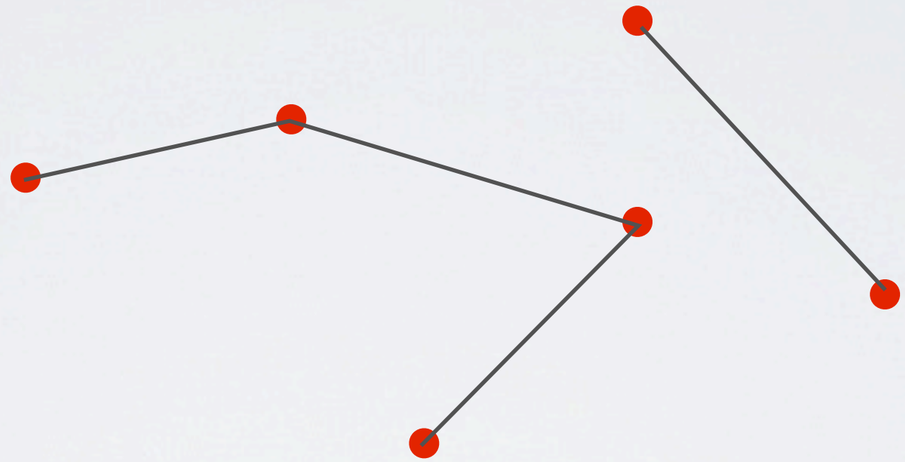
$((x_1, y_1), (x_2, y_2), (x_3, y_3))$

LineStringField



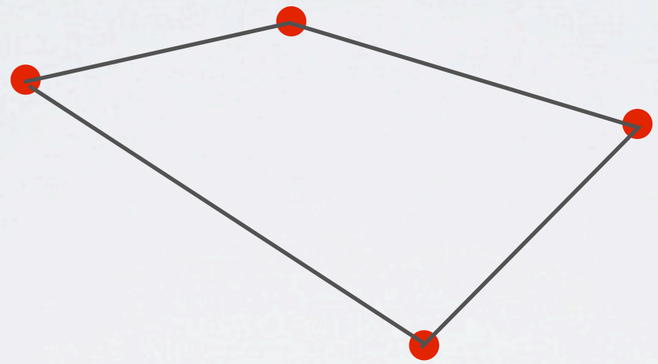
((x1, y1), (x2, y2), (x3, y3), (x4, y4))

MultiLineStringField



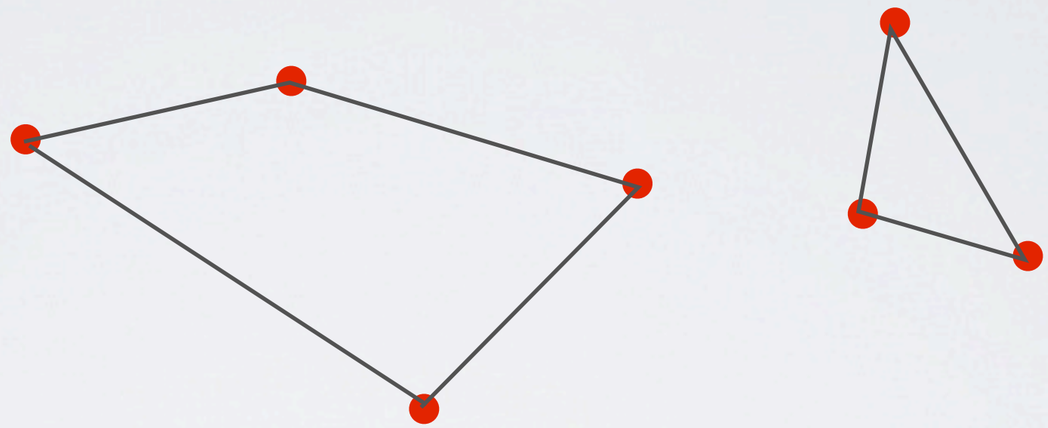
$$(((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)),$$
$$((x_5, y_5), (x_6, y_6)))$$

PolygonField



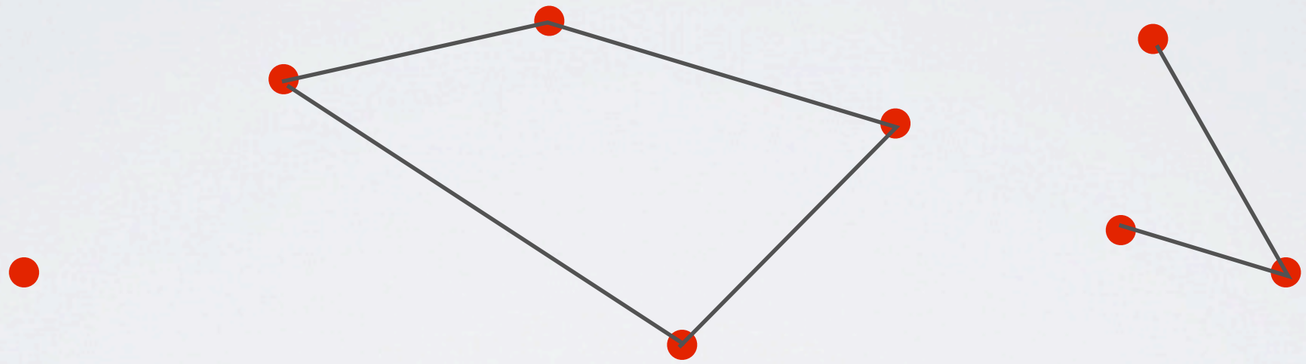
$((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_1, y_1))$

MultiPolygonField



$$((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_1, y_1)),$$
$$((x_5, y_5), (x_6, y_6), (x_7, y_7), (x_5, y_5)))$$

GeometryCollectionField



$$((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_1, y_1)),$$
$$((x_5, y_5), (x_6, y_6), (x_7, y_7)),$$
$$((x_8, y_8)))$$

Practical Example #1: Species Extinction Calculator

Practical Example #2: REDDCalculator.com

```
from django.contrib.gis.gdal import datasource, spatialreference
from django.contrib.gis.geos import multipolygon, geosgeometry

shapefile = os.path.abspath(os.path.join(os.path.dirname(__file__),
'../../../../proj_public/media/files/giveshapefile.shp'))

ds = DataSource(shapefile)

layer = ds[0]   # Checked in Python shell that this has only 1 layer
```

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

class AreaData(models.Model):
    district = models.ForeignKey(District)  # District is FKed to Province
    land_use = models.IntegerField(choices=VEGETATION_TYPES)
    allocation = models.IntegerField(choices=\
                                        CHOICES['LAND_ALLOCATION'])
    intact = models.IntegerField()
    concession = models.CharField(max_length=255, blank=True)
    peat = models.CharField(max_length=255, blank=True)
    area = models.FloatField(help_text='hectares')
    fid = models.IntegerField()
    # ... and any other.
    geom = models.MultiPolygonField(srid=4326, null=True)
    objects = models.GeoManager()
```

```
area = AreaData.objects.get(fid = feature.fid)
geometry_name = feature.geom.geom_type.name
geometry_object = feature.geom.transform(SpatialReference(4326),
clone=True).geos

# update our area object with the geom data from our shape file
if geometry_name == 'Polygon':
    # If our geometry object is a Polygon, we cast as MultiPolygon first
    geometry_object = MultiPolygon(geometry_object)

area.geom = geometry_object
area.save()
```

Geospatial Libraries

GEOS

GDAL

PROJ.4

Mapnik

GeoIP

GEOS

Geometry Engine - Open Source

What does it do?

- Works with django GIS app to give us GeoModels!

GEOS

Geometry Engine - Open Source

Other Details

- Required dependency for django GIS app
- C++
- Implements OGC simple feature for SQL specification
- OGC = Open Geospatial Consortium
- e.g. sudo port install geos

GDAL

Geospatial Data Abstraction Library

What does it do?

- Reads and writes to a variety of vector file formats, e.g. shp

GDAL

Geospatial Data Abstraction Library

Other Details

- C/C++
- Implements OGC simple feature specification
- e.g. sudo port install gdal

PROJ.4

Cartographic Projections Library

What does it do?

- Manages spatial reference systems and projections e.g. the mercator projection commonly used by Google Maps, MapQuest etc

PROJ.4

Cartographic Projections Library

Other Details

- Required dependency for PostGIS
- C
- Ensures that geometry objects stored in PostGIS use the appropriate projection
- sudo port install libproj4

Mapnik

Toolkit for developing Mapping Applications

What does it do?

- For the control freak in you :-)
- Build your own maps, control and customize every single detail

Mapnik

Toolkit for developing Mapping Applications

Other Details

- C++
- Uses boost for python interface
- No py27-mapnik yet. Solution? Create your own local portfile using py26-mapnik
- sudo port install py27-mapnik

GeoIP

Locate users by IP address

What does it do?

- Determines the location of app user via user's IP address

GeoIP

Locate users by IP address

Other Details

- C
- No py27-geoip yet. Solution? Create your own local portfile using py26-geoip
- sudo port install libgeoip py27-libgeoip

Utilities

LayerMapping

./manage.py ogrinspect

./manage.py ogrinfo

./manage.py ogrinspect

and

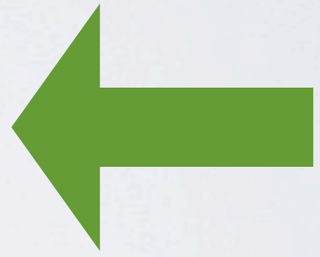
LayerMapping

$ ./manage.py ogrinspect ../proj_public/media/files/redd_data/kalimantan_universe.shp Location --srid=4326 --mapping

```
# This is an auto-generated Django model module created by ogrinspect.
from django.contrib.gis.db import models
class Location(models.Model):
    island = models.CharField(max_length=50)
    geom = models.MultiPolygonField(srid=4326)
    objects = models.GeoManager()

# Auto-generated `LayerMapping` dictionary for Location model
location_mapping = {
    'island' : 'island',
    'geom' : 'MULTIPOLYGON',
}
```

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

class Location(models.Model):
    island = models.CharField(max_length=50)
    geom = models.MultiPolygonField(srid=4326)
    objects = models.GeoManager()

# Auto-generated `LayerMapping` dictionary for Kalimantan model
location_mapping = {
    'island' : 'island',
    'geom' : 'MULTIPOLYGON',
}
```

```
import os
from django.contrib.gis.utils import LayerMapping
from django.contrib.gis.gdal import DataSource
from location.models import Location, location_mapping

shapefile = os.path.abspath(os.path.join(os.path.dirname(__file__),
'../../../../proj_public/media/files/kalimantan_universe.shp'))
ds = DataSource(shapefile)
layer = ds[0]   # Checked in Python shell that this has only 1 layer
kalimantan = LayerMapping(Location,
                          shapefile,
                          location_mapping,
                          transform = True,
                          encoding='iso-8859-1')
kalimantan.save(strict = True, verbose = True)
```




GeoAdmin

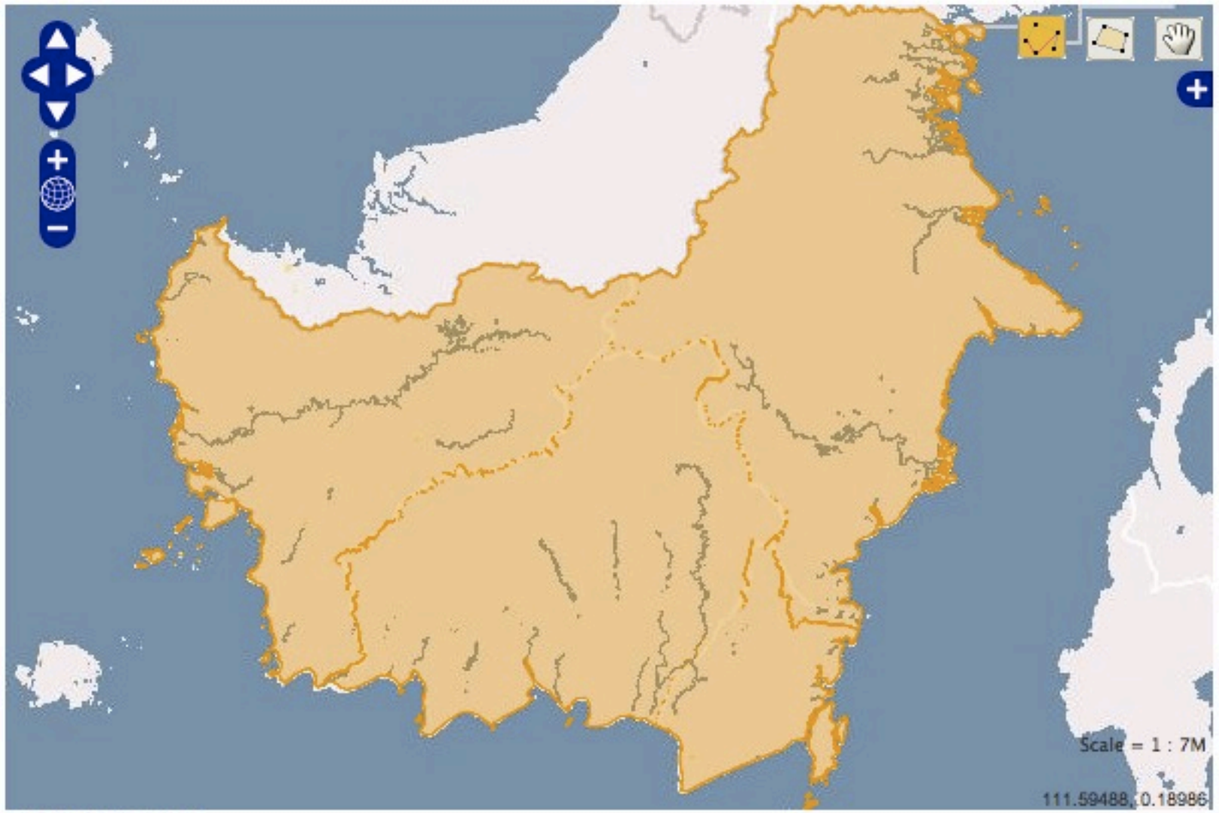
```
from django.contrib.gis import admin as geoadmin
from location.models import Location

admin.site.register(Location, geoadmin.GeoModelAdmin)
```

Change location

History

Island: borneo

Geom:

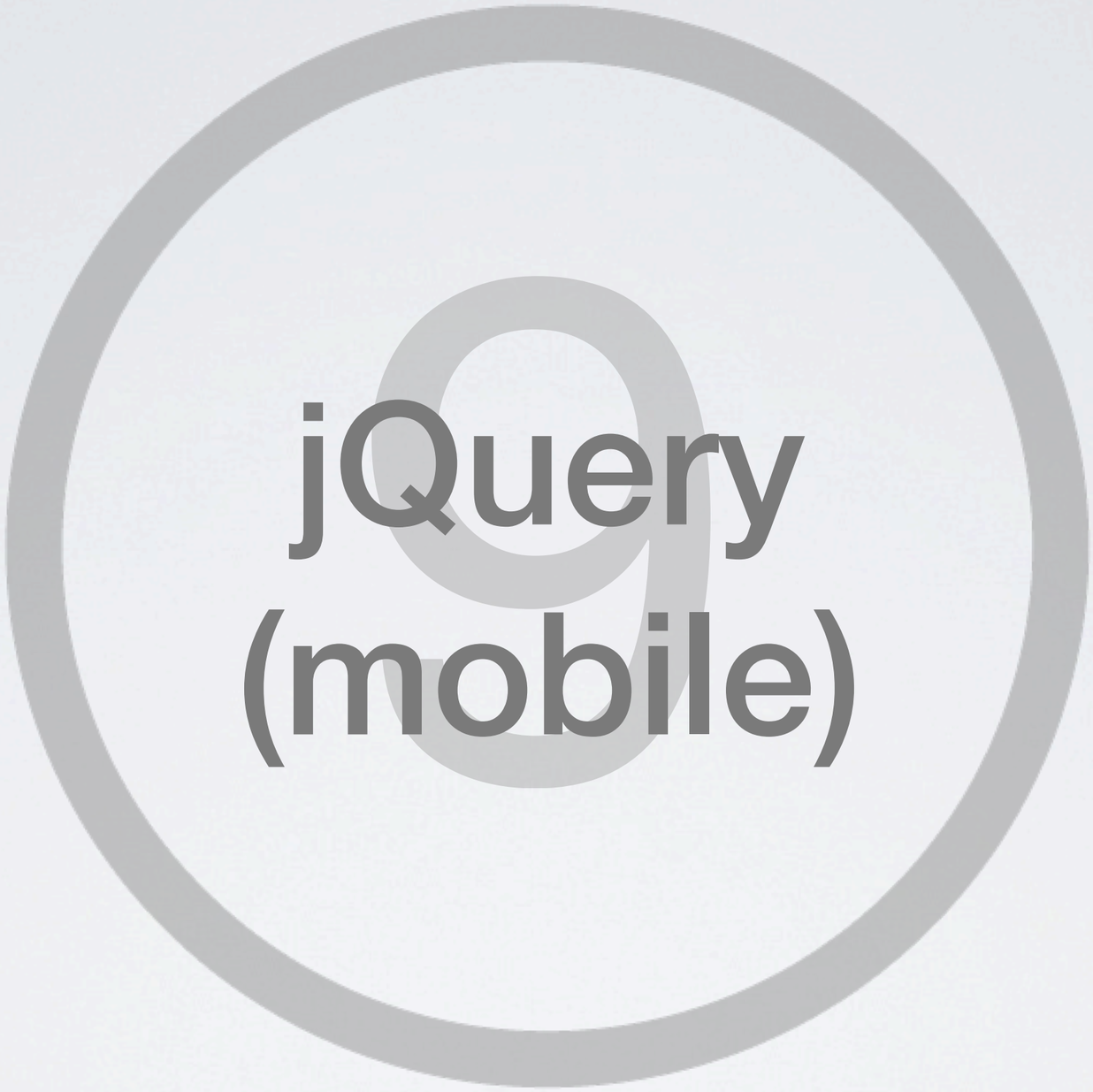


Delete all Features

Delete | Save and add another | Save and continue editing | Save


OpenLayers

“ OpenLayers is a pure JavaScript library for displaying map data in most modern web browsers, with no server-side dependencies. ”


jQuery
(mobile)

“ jQuery Mobile is a Touch-Optimized Web Framework for Smartphones and Tablets. ”

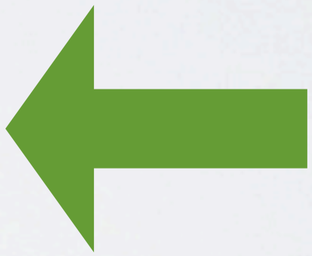

jQuery
mobile framework
+
+
django

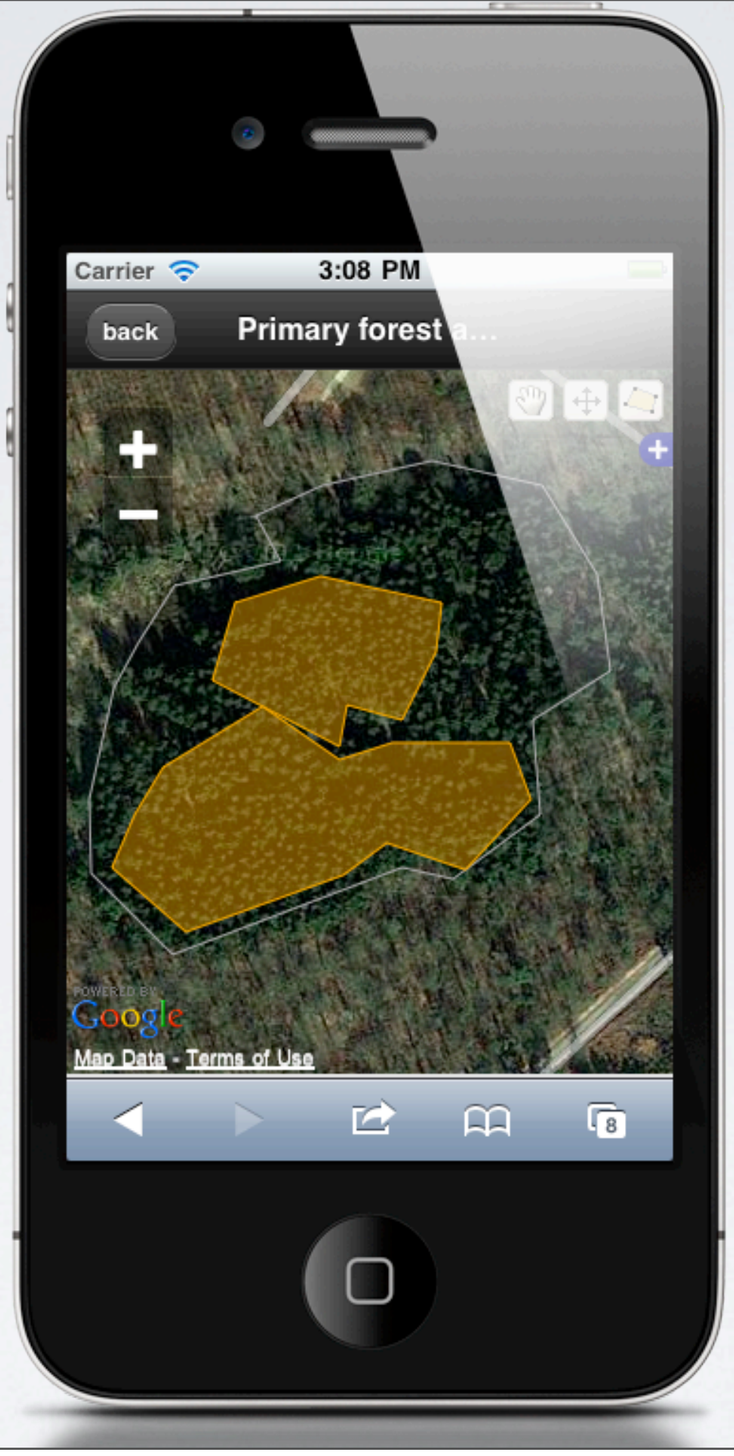
```
$(document).ready(function(){
    $('#map_delete').hide()

    // Define a custom styles which will be used by our polygon layer
    var style = $.extend(true, {}, OpenLayers.Feature.Vector.style['default']) // get a copy of default style
    style.pointRadius = 15
    var styleMap = new OpenLayers.StyleMap({"default": style})

    // Define a custom style which will be used by our background layer
    var backgroundStyle = $.extend(true, {}, OpenLayers.Feature.Vector.style['default']) // get copy
    backgroundStyle.fillOpacity = "0.1"
    backgroundStyle.fillColor = "#000000"
    backgroundStyle.strokeColor = "#999999"
    var backgroundStyleMap = new OpenLayers.StyleMap({"default": backgroundStyle})

    // Define our background layer
    backgroundLayer = new OpenLayers.Layer.Vector("Background Layer", {styleMap: backgroundStyleMap} )
```

```
// Define our vector layer
polygonLayer = new OpenLayers.Layer.Vector("Polygon Layer", {styleMap: styleMap})
polygonLayer.events.on({
    'featureselected': show_delete_button,
    'featureunselected': hide_delete_button
})

// Define our toolbar, panels and associated controls
// pinch and zoom is defined in touch navigation
var touchNavigation = new OpenLayers.Control.TouchNavigation({
    dragPanOptions: {
        interval: 100,
        enableKinetic: true
    }
})

// define our zoom panel
var zoomPanel = new OpenLayers.Control.ZoomPanel()

// define our layer switcher
var layerSwitcher = new OpenLayers.Control.LayerSwitcher()
```

```
// controls
drawControl = new OpenLayers.Control.DrawFeature(polygonLayer, OpenLayers.Handler.Polygon,
                                                 { displayClass: 'olControlDrawFeaturePolygon'});
selectControl = new OpenLayers.Control.SelectFeature(polygonLayer,
                                                 { clickout: true });
editControl = new OpenLayers.Control.ModifyFeature(polygonLayer,
                                                 { selectControl: selectControl,
                                                   vertexRenderIntent: 'temporary',
                                                   displayClass: 'olControlModifyFeature'});

// define our custom control tool bar
var toolbar = new OpenLayers.Control.Panel({displayClass: 'olControlEditingToolbar'});
toolbar.addControls([
    //this control is just there to be able to deactivate the drawing tools and pan the map
    drawControl,
    editControl,
    new OpenLayers.Control({ displayClass: 'olControlNavigation' })
]);
```

```
// Define Map Layers
var gphy = new OpenLayers.Layer.Google("Google Physical", {type: google.maps.MapTypeId.TERRAIN});
var gmap = new OpenLayers.Layer.Google("Google Streets", {numZoomLevels: 20});
var ghyb = new OpenLayers.Layer.Google("Google Hybrid", {type: google.maps.MapTypeId.HYBRID,
                                                         numZoomLevels: 20});
var gsat = new OpenLayers.Layer.Google("Google Satellite", {type: google.maps.MapTypeId.SATELLITE,
                                                         numZoomLevels: 20});

// Feed the controls and layers defined above to our map object
map = new OpenLayers.Map({
    div: 'map',
    controls: [zoomPanel, touchNavigation, layerSwitcher, toolbar],
    layers: [ghyb, gphy, gmap, gsat, backgroundLayer, polygonLayer]
})
```




Carrier
3:08 PM
back
Primary forest a…
+
−
POWERED BY
Google
Map Data - Terms of Use
8

```
    // Google.v3 uses EPSG:900913 as projection, so we have to transform our coordinates
    mapProjection = map.getProjectionObject()
    map.setCenter(new OpenLayers.LonLat(map_center_lon, map_center_lat).transform(defaultProjection,
mapProjection), map_zoom)

    // if there's a Total Landscape Area polygon, we will draw it on the backgroundLayer.
    if (background_coords_json){
        var pointList = []
        for (var i=0; i<background_coords_json.length; i++) {
            var point = new OpenLayers.Geometry.Point(background_coords_json[i][0], background_coords_json[i][1])
            pointList.push(point.transform(defaultProjection, mapProjection));
        }
        var linearRing = new OpenLayers.Geometry.LinearRing(pointList)
        totalAreaPolygon = new OpenLayers.Feature.Vector(new OpenLayers.Geometry.Polygon([linearRing]))
        backgroundLayer.addFeatures([totalAreaPolygon])
    }
```

```
// draw the existing features into the polygonLayer
var loaded_features = []
for (var i=0; i<list_of_coords_json.length; i++) {
    var pointList = []
    for (var j=0; j<list_of_coords_json[i].length; j++) {
        var point = new OpenLayers.Geometry.Point(list_of_coords_json[i][j][0], list_of_coords_json[i][j][1])
       pointList.push(point.transform(defaultProjection, mapProjection));
    }
    var linearRing = new OpenLayers.Geometry.LinearRing(pointList)
    var polygonFeature = new OpenLayers.Feature.Vector(new OpenLayers.Geometry.Polygon([linearRing]),
                                                       {fid: list_of_coords_ids_json[i]})
    loaded_features.push(polygonFeature)
    }
    polygonLayer.addFeatures(loaded_features)
    polygonLayer.events.on({ "featuremodified": polygonSave, "featureadded": polygonSave,
         "vertexmodified": adjustVertex, "sketchstarted": adjustVertex, "sketchmodified": adjustVertex })

    // confirmation dialog
    $('#project_map_delete_polygon_yes').click(delete_polygon)
})
/* END of $(document).ready() */
```

Possibilities

Offline Maps for Field Usage

Native
iOS/Android

Layers & Features (Saved Locally, Syncing)

Map Source (Local Map Tiles)

Native?

KML | GML | WKT | GeoJSON | GeoRSS | Others

Data Transfer

GeoDjango

Python Logic

GDAL

OSGeo
Your Open Source Compass

mapnik

PROJ.4

PostgreSQL

PostGIS
Geospatial Objects for PostgreSQL

C/C++ Libraries
w/ Python Bindings
& Datastore

The Problem with iOS MapKit Framework

- MKMapView class implements Google Maps API natively
- Unfortunately, Google Maps terms of use prevents us from caching or storing map tiles locally

Alternative? https://github.com/route-me/route-me

- Supports many map sources - OpenStreetMap, Microsoft VirtualEarth, CloudMade, OpenAerialMap, OpenCycleMap, SpatialCloud
- OpenStreetMap permits map tile caching and local storage
- Supports 2 offline, database-backed formats DBMap and MBTiles

A Comparison

	MKMapKit	Route-Me
Embed in iOS App	YES	YES
Interaction	YES	YES
Geolocalization	YES	YES
Different Map Sources	NO	YES
Offline Maps	NO	YES

The Problem with Android Google Maps Framework

- Renders Google Maps of course
- Support for alternative map sources is important

Alternative? https://github.com/camptocamp/android-gis.git

- Supports many map sources - OpenStreetMap, Microsoft VirtualEarth, CloudMade, OpenAerialMap, OpenCycleMap, SpatialCloud
- OpenStreetMap permits map tile caching and local storage
- Supports 2 offline, database-backed formats DBMap and MBTiles

Benefits

- Fast
  (obviously, since Map Tiles are stored locally)
- 100% Offline
- Stored as blob data for SQLite Client Access

Getting a GPS fix via Satellite?

11 Conclusion

Mathematical Modelling, Scenario Analysis for Environmental Policies

Koh_et_al_moratorium_tool.xlsm

Home | Layout | Tables | Charts | SmartArt | Formulas | Data | Review | Developer

L5

Spatially Explicit Scenario Analysis of the Norway-Indonesia REDD+ Moratorium: Environmental and socioeconomic tradeoffs in Kalimantan, Indonesia

Lian Pin Koh (ETH Zurich), Holly Gibbs (Univ. Wisconsin-Madison), Matthew Hansen (South Dakota State Univ.)

Model Scenario *(scope of moratorium)*

Land Allocation

- [ ] Protected Area
- [ ] Natural Protected Forests
- [x] Forest Limited Production
- [x] Forest Production Conversion
- [x] Production Forest
- [x] Non-Forest Land

Concessions

- [ ] Logging Concessions
- [ ] Timber Plantations
- [ ] Tree Crop Plantations

Land Use

'Primary/Intact Forests':

- [x] Lowland Forest
- [x] Peatswamp Forest
- [x] Mangrove
- [x] Lower Montane Forest
- [x] Upper Montane Forest

'Closed-Canopy Secondary Forests':

- [x] Plantation/Regrowth

'Open vegetation':

- [ ] Lowland Mosaic
- [ ] Montane Mosaic
- [ ] Lowland Open
- [ ] Montane Open

Biophysical constraints

Biomass carbon: Unconstrained

Peatland peat depth: Unconstrained

Forest 'intactness': Unconstrained

Save

Print

Update model

Summary Outcome *(see Tables 1 and 2 for detailed results)*

Display: Entire Kalimantan

Total land area (thousand ha)

Included	Excluded
18,703	33,177

0% 20% 40% 60% 80% 100%

Biomass and peat carbon (thousand Mg)

Conserved	Not conserved
9,723,162	9,474,799

0% 20% 40% 60% 80% 100%

Potential financial returns, NPV* (million $)

	Retained
	227,966

Model Assumptions *(can be modified as necessary)*

Biomass Carbon (Mg/ha) (Note that peat carbon is accounted for in the model, but not reflected in this table.)

Vegetation Type	Aboveground	Belowground (root)	Total
Mangrove	159	31.8	190.8
Peatswamp forest	180	36	216
Lowland forest	225	45	270
Lower montane forest	256	51.2	307.2
Upper montane forest	203	40.6	243.6
Plantation/regrowth	60	12	72
Lowland mosaic	39	7.8	46.8
Montane mosaic	51	10.2	61.2
Lowland open	19.5	3.9	23.4
Montane open	25.5	5.1	30.6

Opportunity cost calculations

Vegetation Type	Timber value ($/ha)	Establishment cost ($/ha)	Operations cost ($/ha/y)	Oil palm CPO yield[1] (t/ha)
Mangrove	4,000.00	5,000.00	1,000.00	4.00
Peatswamp forest	4,000.00	5,000.00	1,000.00	4.00
Lowland forest	10,000.00	4,000.00	1,000.00	5.00
Lower montane forest	10,000.00	4,000.00	1,000.00	5.00
Upper montane forest	10,000.00	4,000.00	1,000.00	5.00
Plantation/regrowth	2,000.00	4,000.00	1,000.00	5.00
Lowland mosaic	0.00	4,000.00	1,000.00	4.00
Montane mosaic	0.00	4,000.00	1,000.00	4.00
Lowland open	0.00	4,000.00	1,000.00	4.00
Montane open	0.00	4,000.00	1,000.00	4.00

CPO price[1] ($/t): 560.00

Annual discount rate: 10.0%

Employment, oil palm (jobs/ha): 0.10

Update model

Notes: [1]Lifetime-average crude palm oil (CPO) yield. A yield curve will be applied to these values to calculate year-by-year production. [2]Average over 25 year modeling timeframe.

This project was initiated and supported by:

UN-REDD PROGRAMME

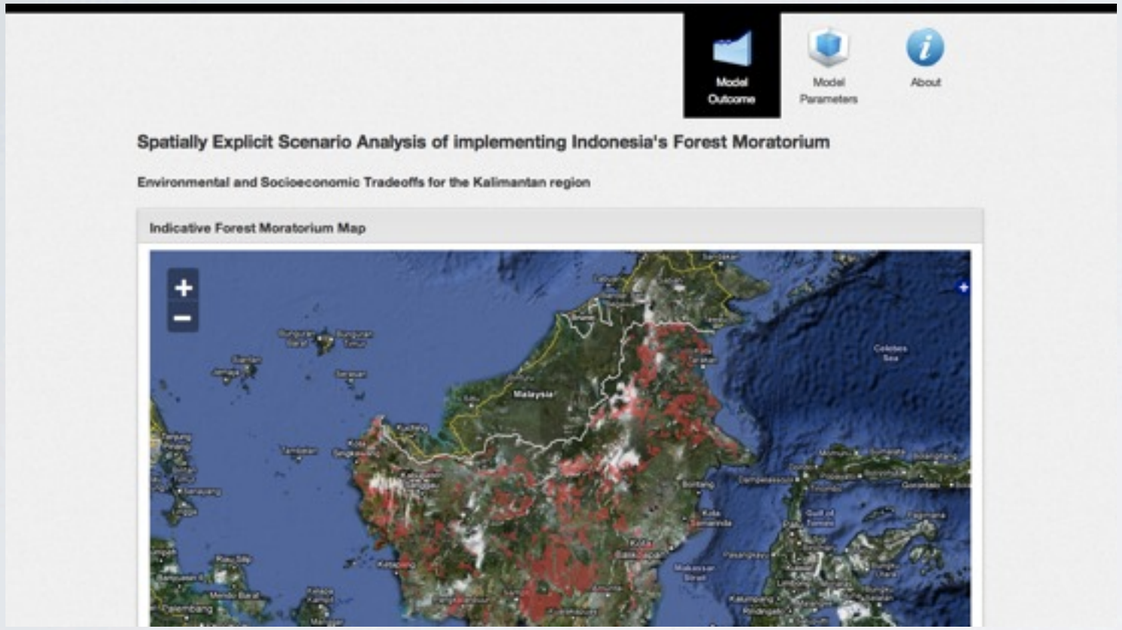
The UN-REDD Programme is the United Nations collaborative initiative on Reducing Emissions from Deforestation and Forest Degradation (REDD) in developing countries. The Programme was launched in 2008 and builds on the convening role and technical expertise of the Food and Agriculture Organization of the United Nations (FAO), the United

Table 1. District-level environmental and socioeconomic implications Indonesia.

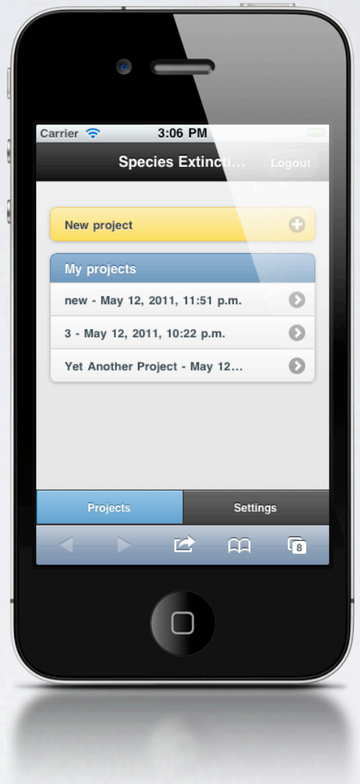
		Environmental benefit
Province	District	Land area (ha)
Barat	Bengkayang	283,814
Barat	Kapuas Hulu	571,117
Barat	Ketapang	1,468,223
Barat	Landak	488,548
Barat	Melawi	815,255
Barat	Pontianak	715,940
Barat	Sambas	318,828
Barat	Sanggau	589,031
Barat	Sekadau	261,093
Barat	Singkawang	29,740
Barat	Sintang	524,494
Selatan	Balangan	73,801
Selatan	Banjar	108,514
Selatan	Banjar Baru	1,740
Selatan	Banjarmasin	502
Selatan	Barito Kuala	48,381
Selatan	Hulu Sungai Selatan	59,887
Selatan	Hulu Sungai Tengah	51,338
Selatan	Hulu Sungai Utara	26,047
Selatan	Kota Baru	180,163
Selatan	Tabalong	120,766
Selatan	Tanah Bumbu	106,307
Selatan	Tanah Laut	69,740
Selatan	Tapin	60,722
Tengah	Barito Selatan	270,745
Tengah	Barito Timur	207,492
Tengah	Barito Utara	443,072
Tengah	Gunung Mas	409,352
Tengah	Kapuas	668,931
Tengah	Katingan	866,677
Tengah	Kotawaringin Barat	355,312
Tengah	Kotawaringin Timur	456,978
Tengah	Lamandau	122,932
Tengah	Murung Raya	841,643
Tengah	Palangka Raya	278,208
Tengah	Pulang Pisau	717,840
Tengah	Seruyan	317,861
Tengah	Sukamara	120,188
Timur	Balikpapan	7,690
Timur	Berau	867,836
Timur	Bontang	18,311
Timur	Bulungan	550,434

Interface | control | data | logic | assumptions | scratch | +

REDDCalculator.com (Beta)



SEC (Closed Beta)



LUC (Coming Soon!)

ENVIRONMENTAL iTOOLS

- Spatially Explicit Scenario Analysis: Norway-Indonesia's Forest Moratorium
- Species Extinction Calculator (SEC)
- Land Use Calculator (LUC)

- **Species Extinction Calculator** and **REDDCalculator**:

  Mathematical Models and Formulas by Koh et. al are completely translated into Python

- **Land Use Calculator**:

  Ditto. Coming Soon!

Key Takeaways

- **Mapnik** is useful for rasterizing vector objects when we encounter large datasets (in our case, 200 MB+ of protected areas in Kalimantan).

- **(jQuery Mobile) Web App** can benefit from optimization through django caching. To be done right after EuroPython! And besides jQuery Mobile Beta 1 has just been released on Monday (20 June 2011)! :-)

- Usability and App Responsiveness can possibly be improved, Out-Field/ Offline Benefits attainable, if we move interface elements and Map Tiles into a native app (Android or iOS).

- **geography=True/False in a GeoDjango field**
- 2 ways to import data from a given GIS Shape File

  (1) LayerMapping

  (2) Direct Assignment to obj.geom attribute as mentioned.

“ Advancing Environmental Sciences:

helping people, scientists, NGOs and governments make practical Land Area Usage and Conservation Decisions using jQuery (Mobile), OpenLayers and GeoDjango! ”

Gotchas?

Make sure Projections/Datum in your shape file source corresponds with your geometry field's SRID and Map Source

Using LayerMapping?

transform = True

Directly creating or saving into an object's attribute?

```
geometry_object = feature.geom.transform(SpatialReference(4326),
clone=True).geos
area.geom = geometry_object
area.save()
```

Credits

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

\+ North-South Fund

Prof. Koh Lian Pin
ETH Zurich, Dept of Env Sciences
NUS, Dept of Biological Sciences

Prof. Matthew Hansen
South Dakota State University
GIS Center of Excellence

Prof. Holly Gibbs
Stanford Dept of Environmental
Earth System Sciences

Odeon
CONSULTING GROUP

Stefan Talpalaru
Odeon Consulting Group

Liviu Bogdan Chiributa
Odeon Consulting Group

That’s Me! :-)
Odeon Consulting Group

Previous GeoDjango Presentations + my little contribution

- www.geodjango.org/presentations

- www.od-eon.com/presentations (Available by tomorrow, 23rd June)

Questions and Discussion welcome!

calvin.cheng@od-eon.com