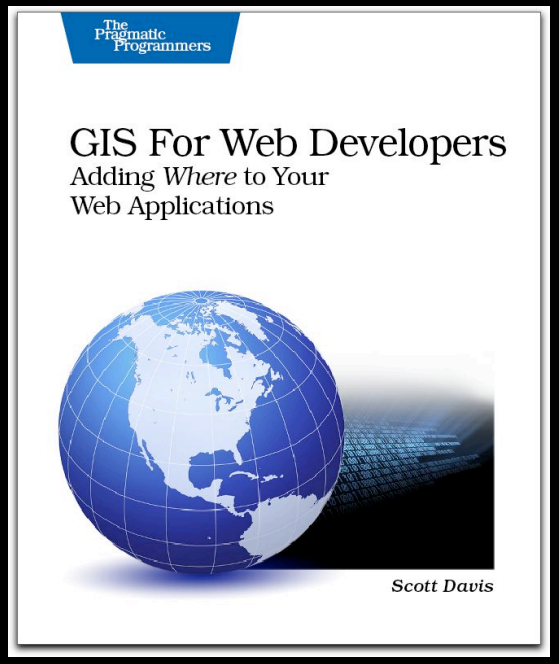
GIS for Web Developers

Adding ***Where*** to Your Application

Introduction

- My name is Scott Davis
  - Editor in Chief of http://aboutGroovy.com
  - Author
    - Groovy Recipes: Greasing the Wheels of Java (Pragmatic Bookshelf)
    - GIS for Web Developers (Pragmatic Bookshelf)
    - Google Maps API (Pragmatic Bookshelf)
    - JBoss At Work (O'Reilly)



Demystifying GIS

- Geographic Information Systems sounds like something you need a degree in
  - (It might help, but it's not a requirement…)

Neo-Geography

- New wave of non-GIS professionals creating maps using free data and open source applications
  - Google Maps, et al

View / Source

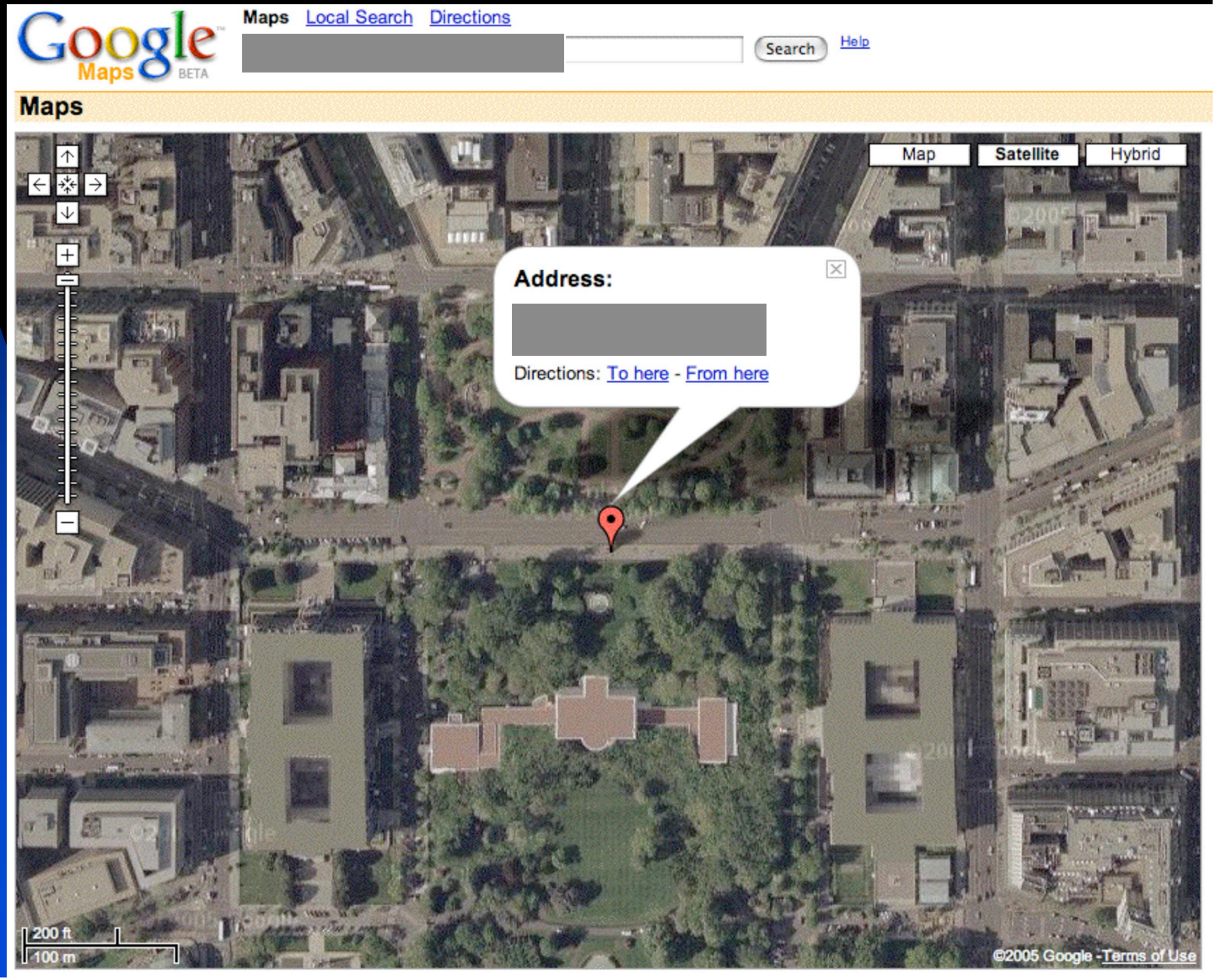
- This talk is the equivalent of doing a View / Source on web maps
  - You will learn the jargon
  - You will find free sources of data
  - You will find free applications to use

Act 1: Free Data

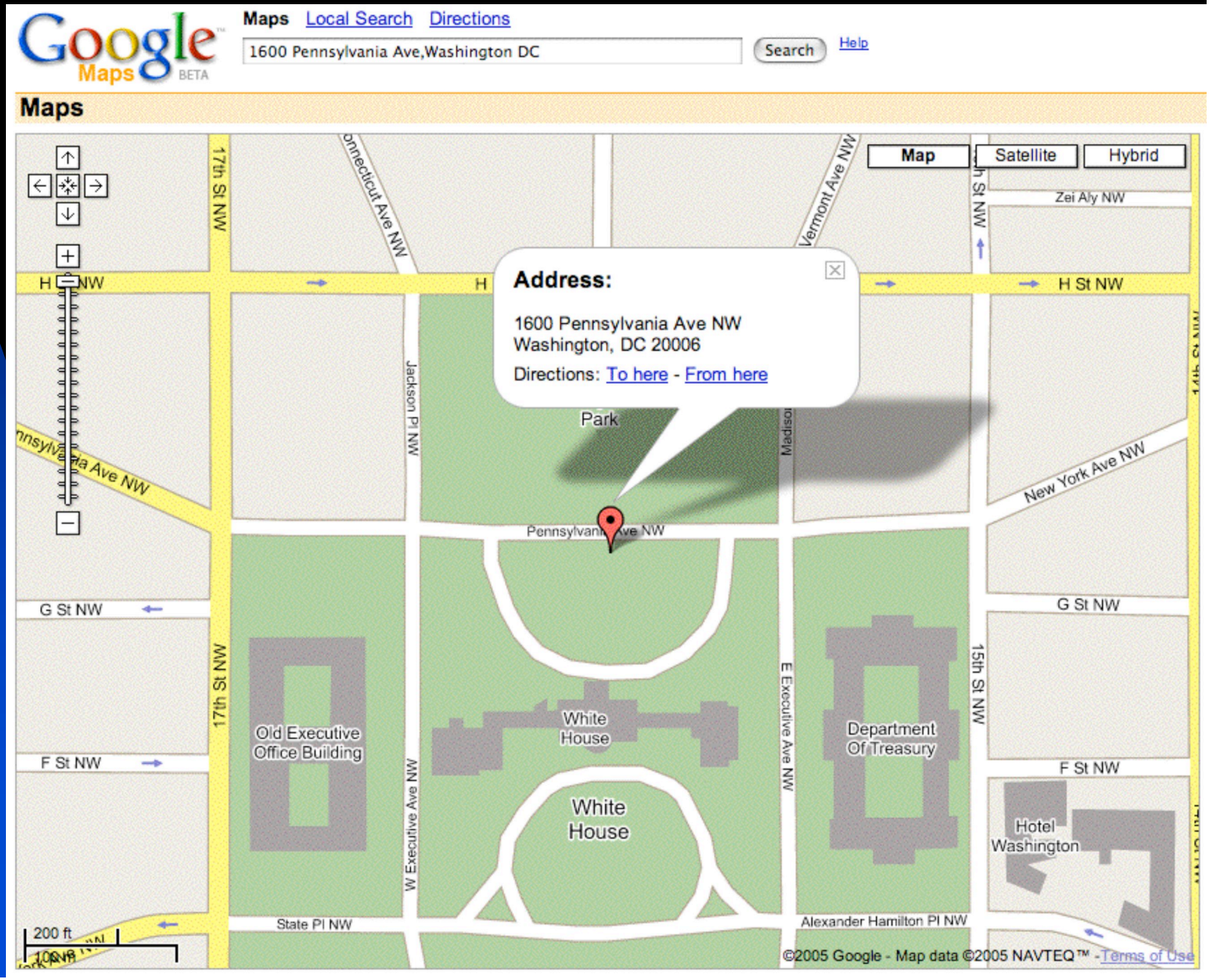
Vectors vs. Rasters

- There are two types of data out there:

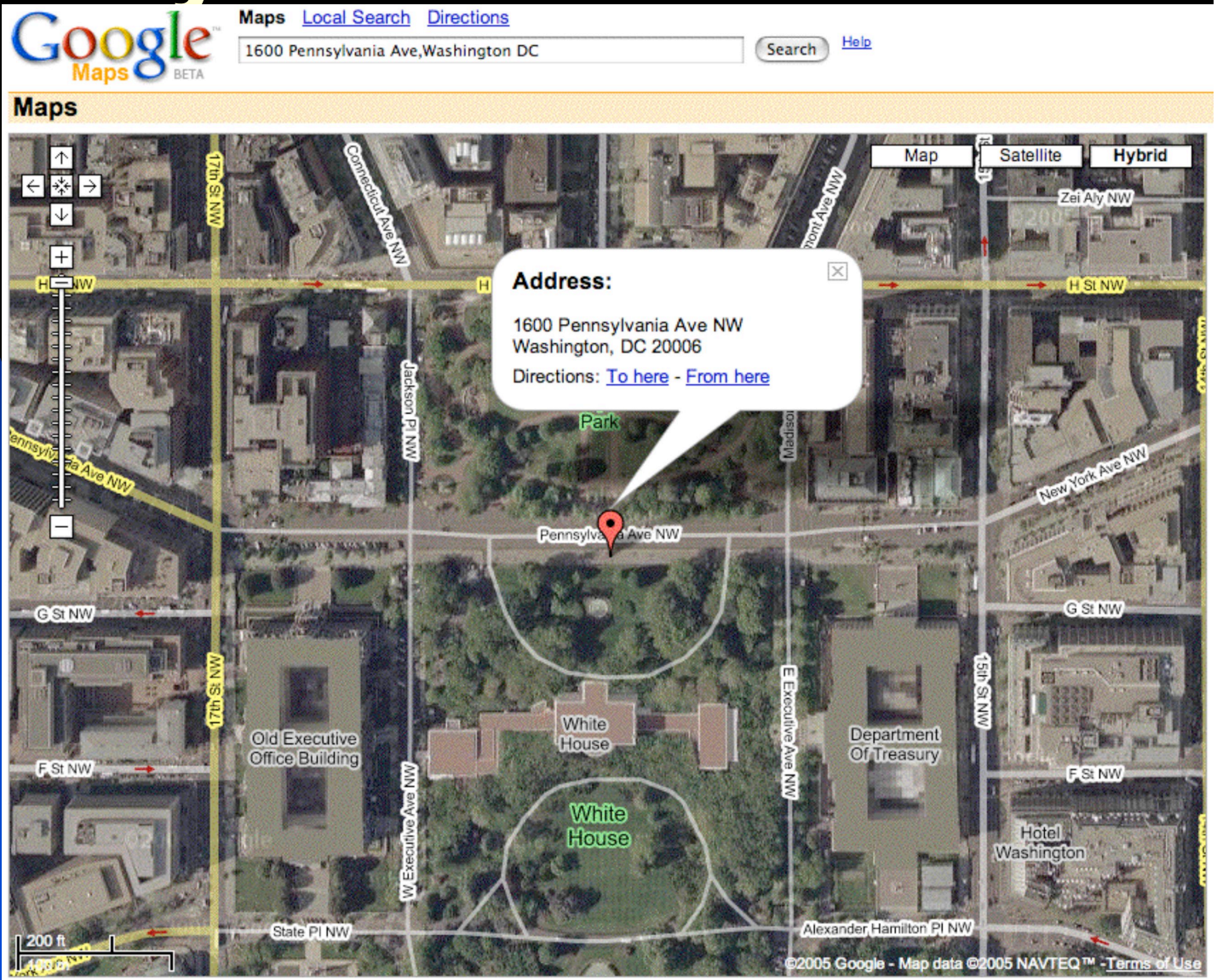
Raster Data



Vector Data



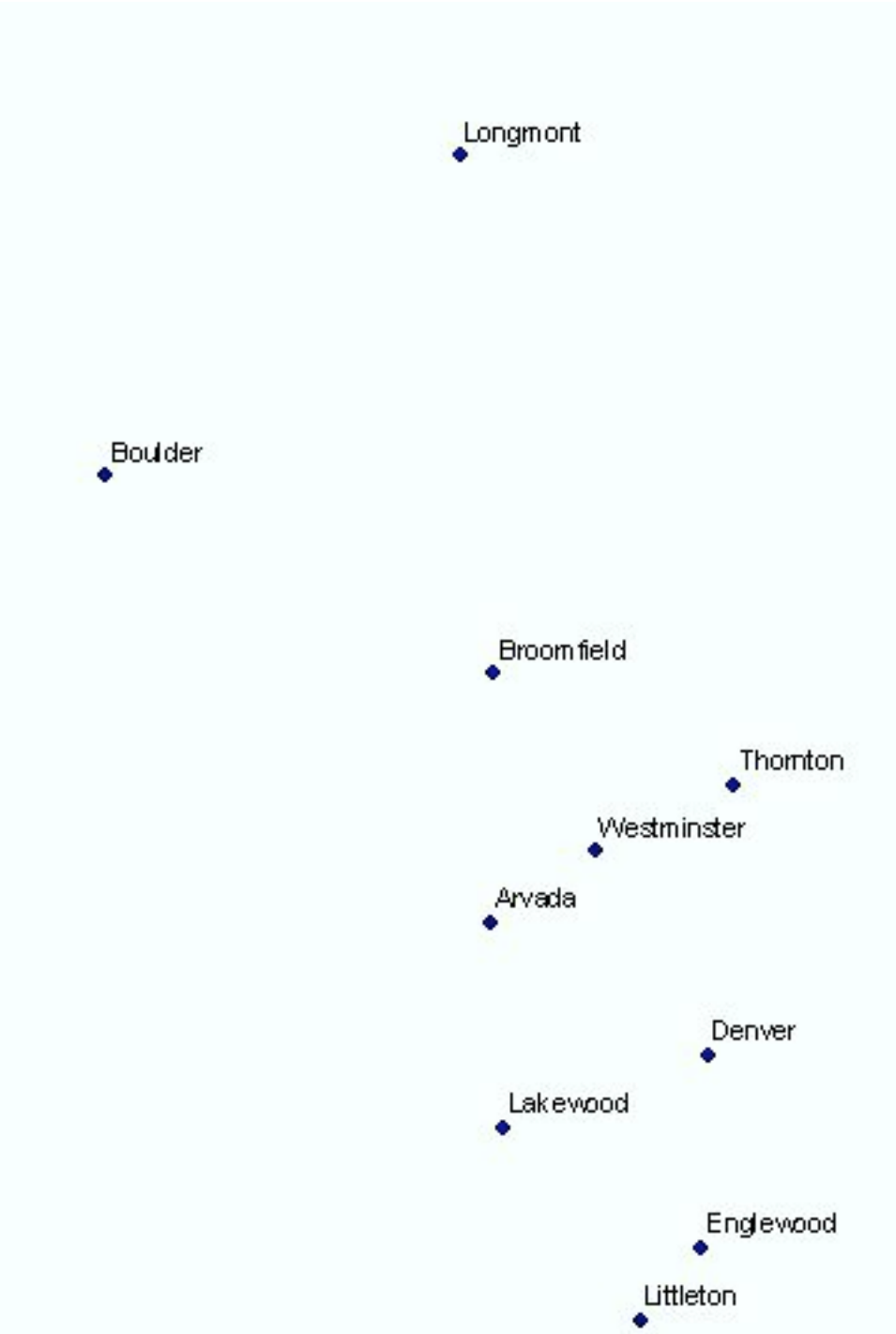
Hybrid



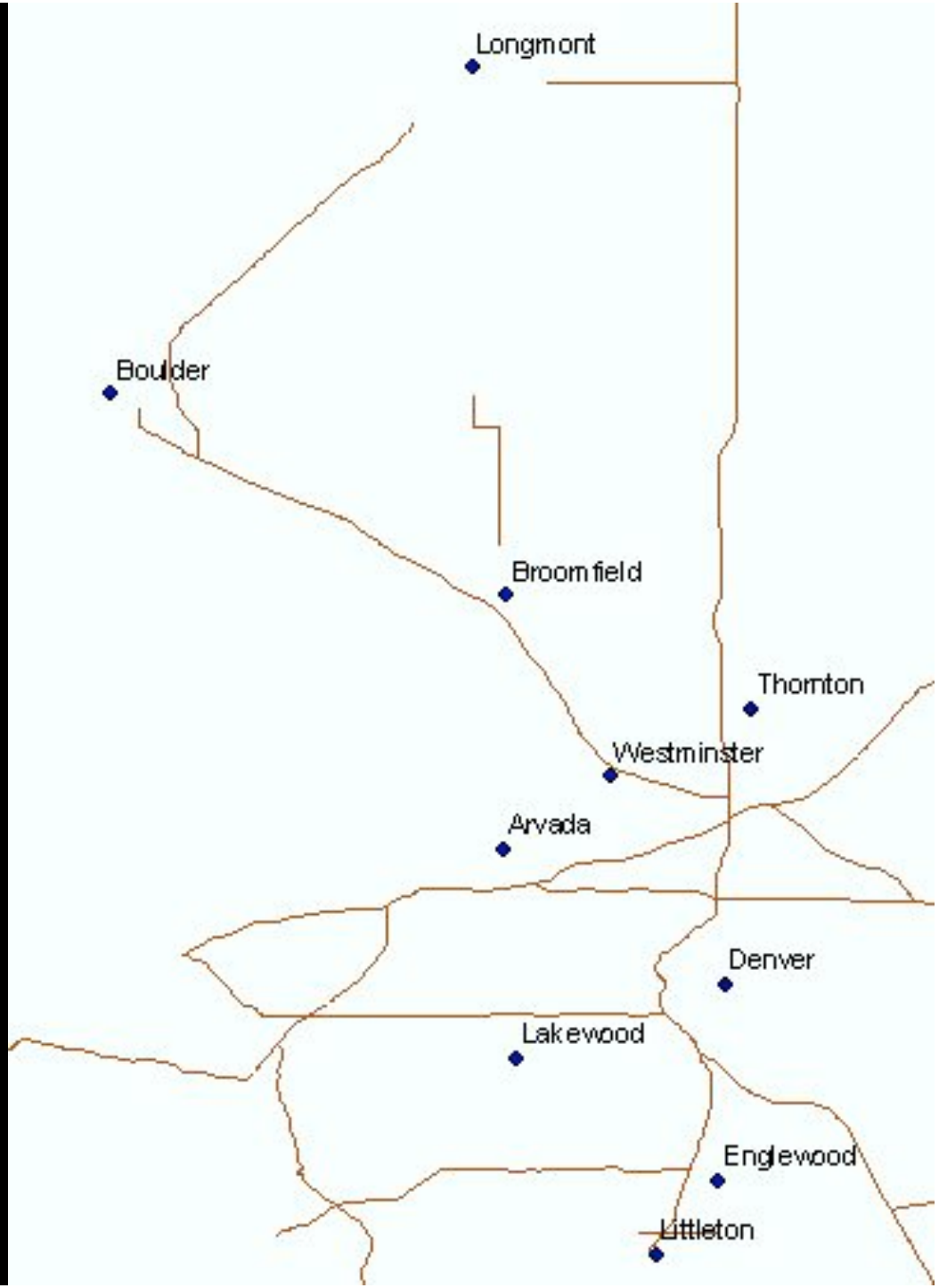
Type of Vectors

- Vectors come in three primitive datatypes:

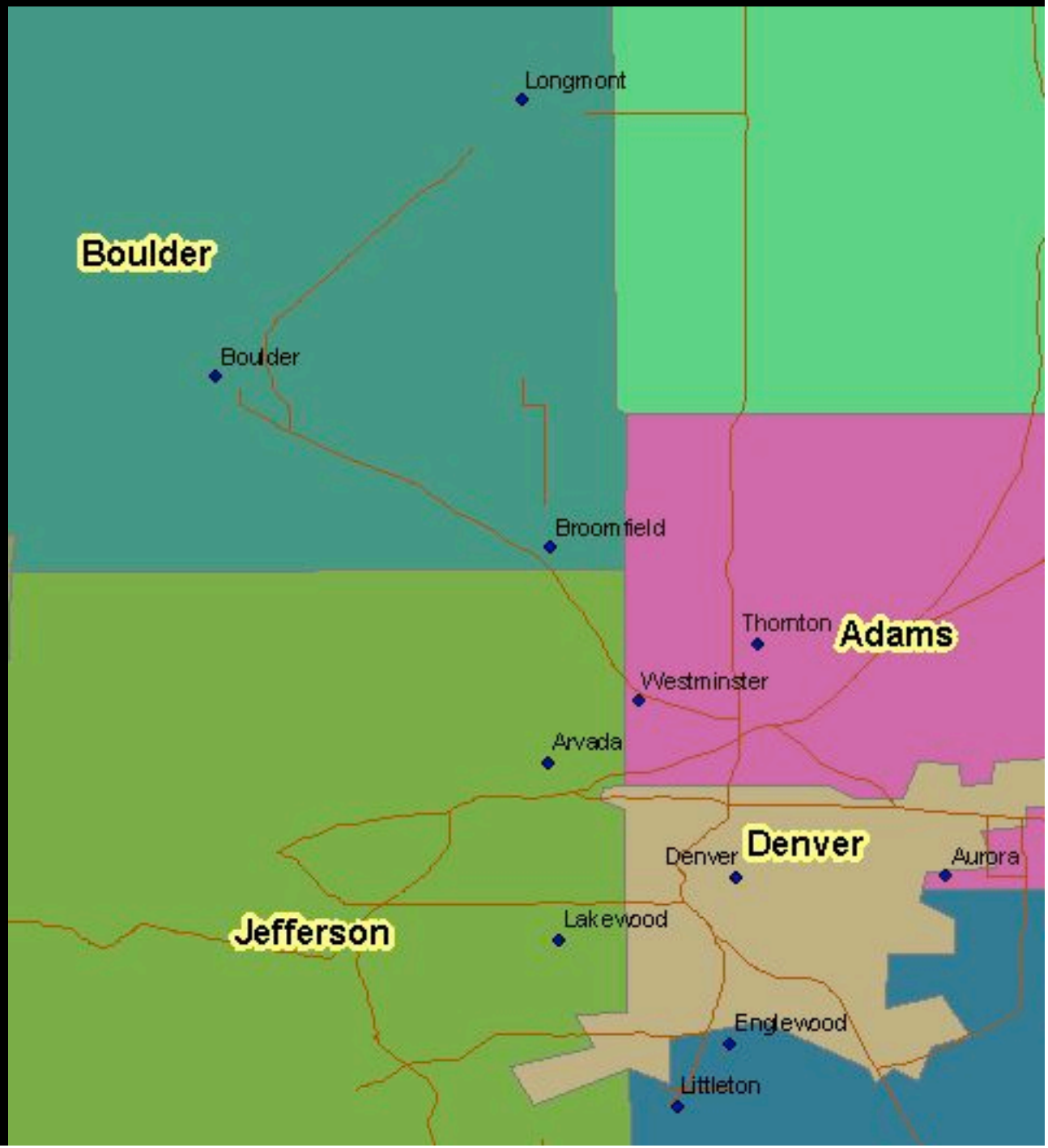
Point



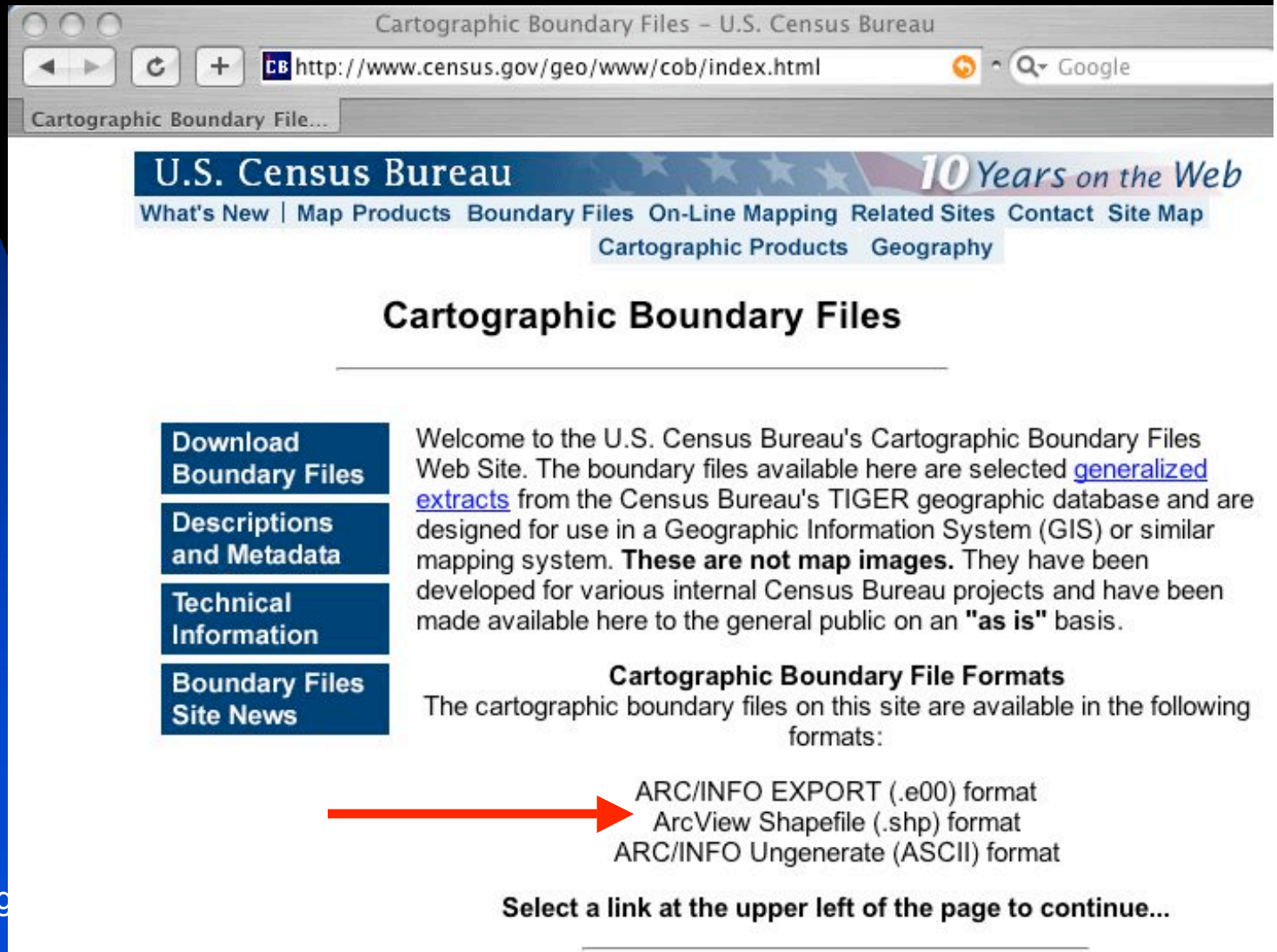
Line



Polygon



Free Vectors:



Shapefile

- Shapefiles are a proprietary ESRI standard, but the format is well documented:
  - http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
- Each shapefile is made up of several files:
  - .shp, .shx, .dbf

The Downloadable States of America

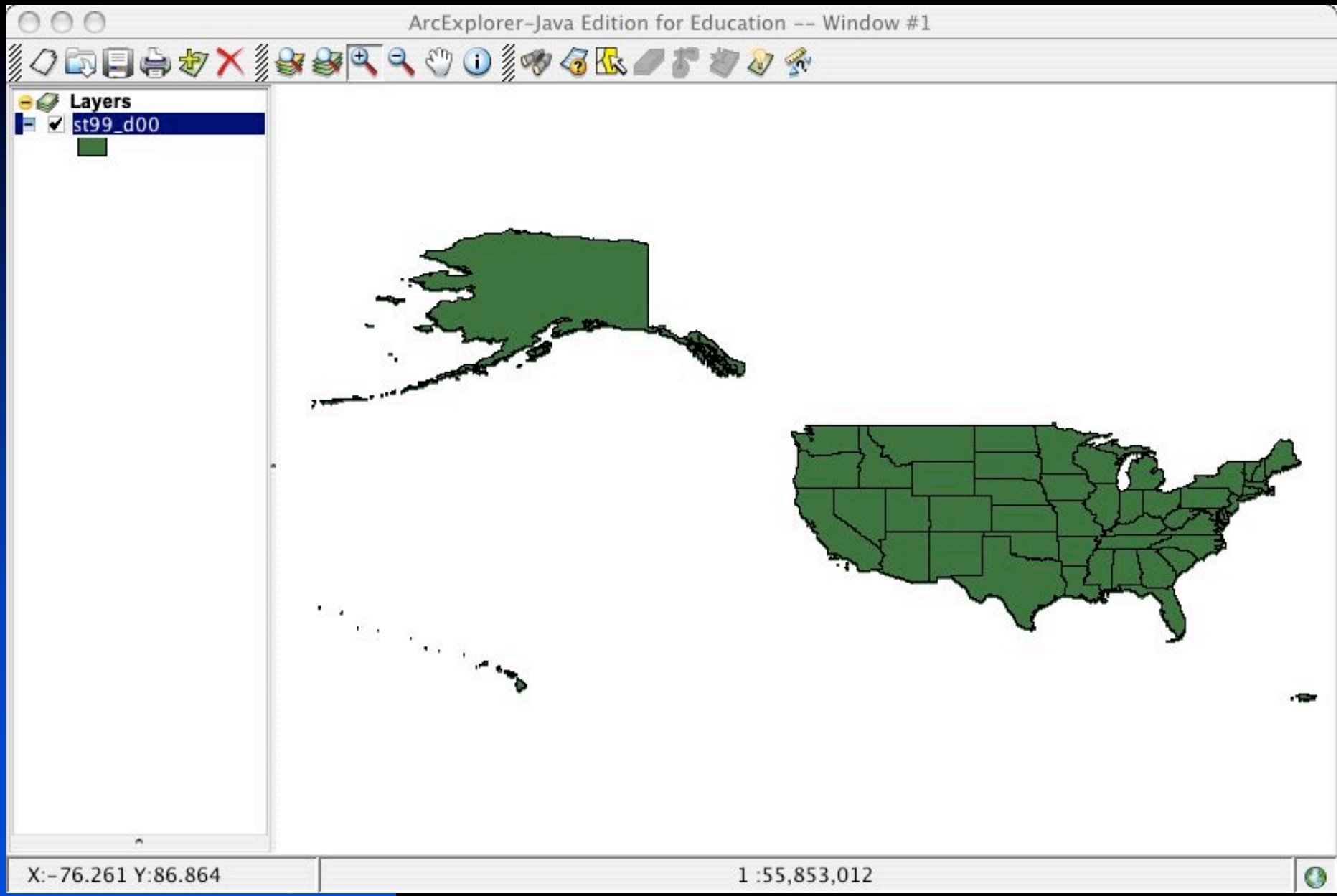
- http://www.census.gov/geo/cob/bdy/st/st00shp/st99_d00_shp.zip

Uh, OK.

Now what?

Downloading Viewers

- ESRI offers a free viewer:
  - http://www.esri.com/software/arcexplorer/download.html

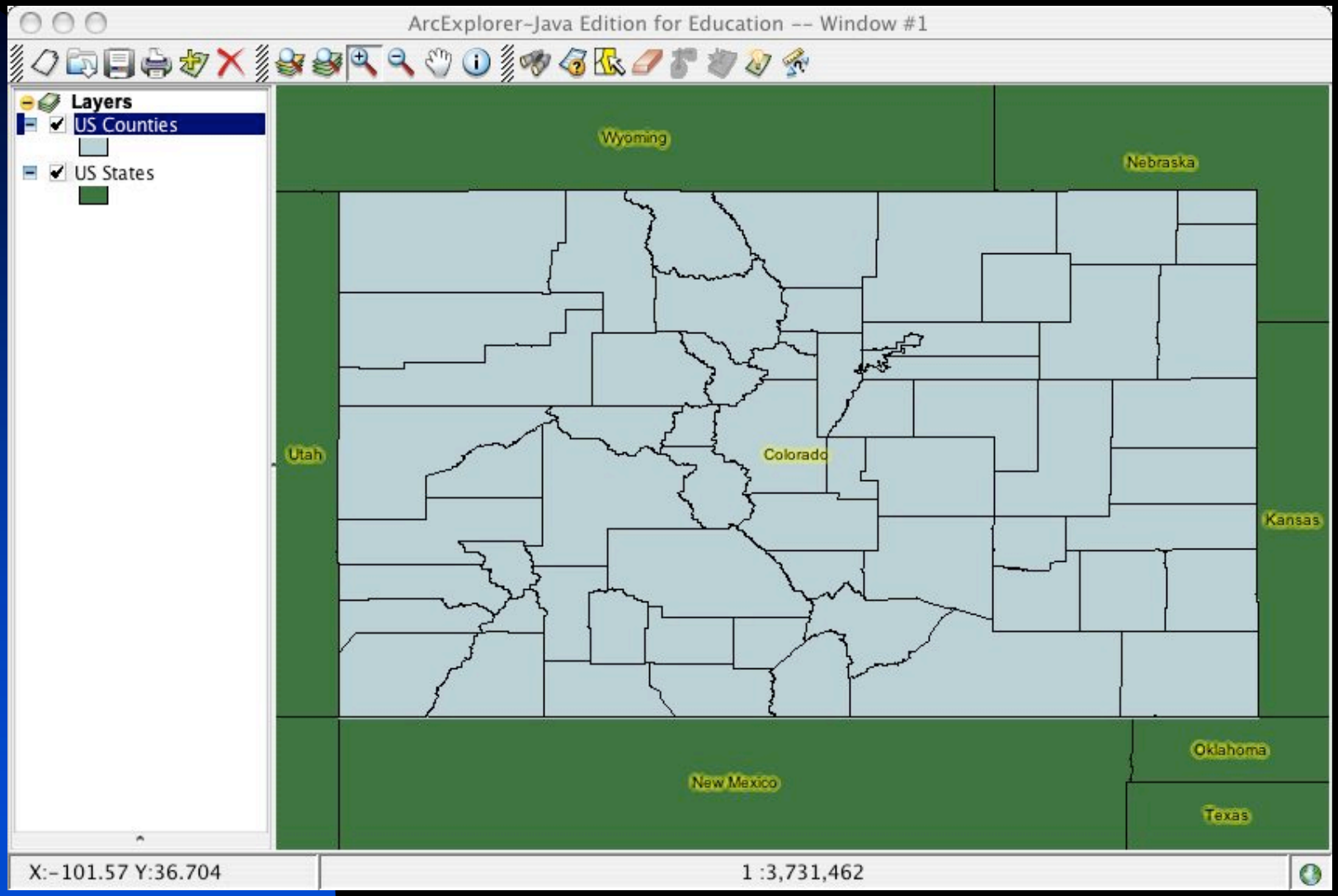

ArcExplorer-Java Edition for Education -- Window #1
Layers
st99_d00
X:-76.261 Y:86.864
1 :55,853,012

Non-Spatial Attributes

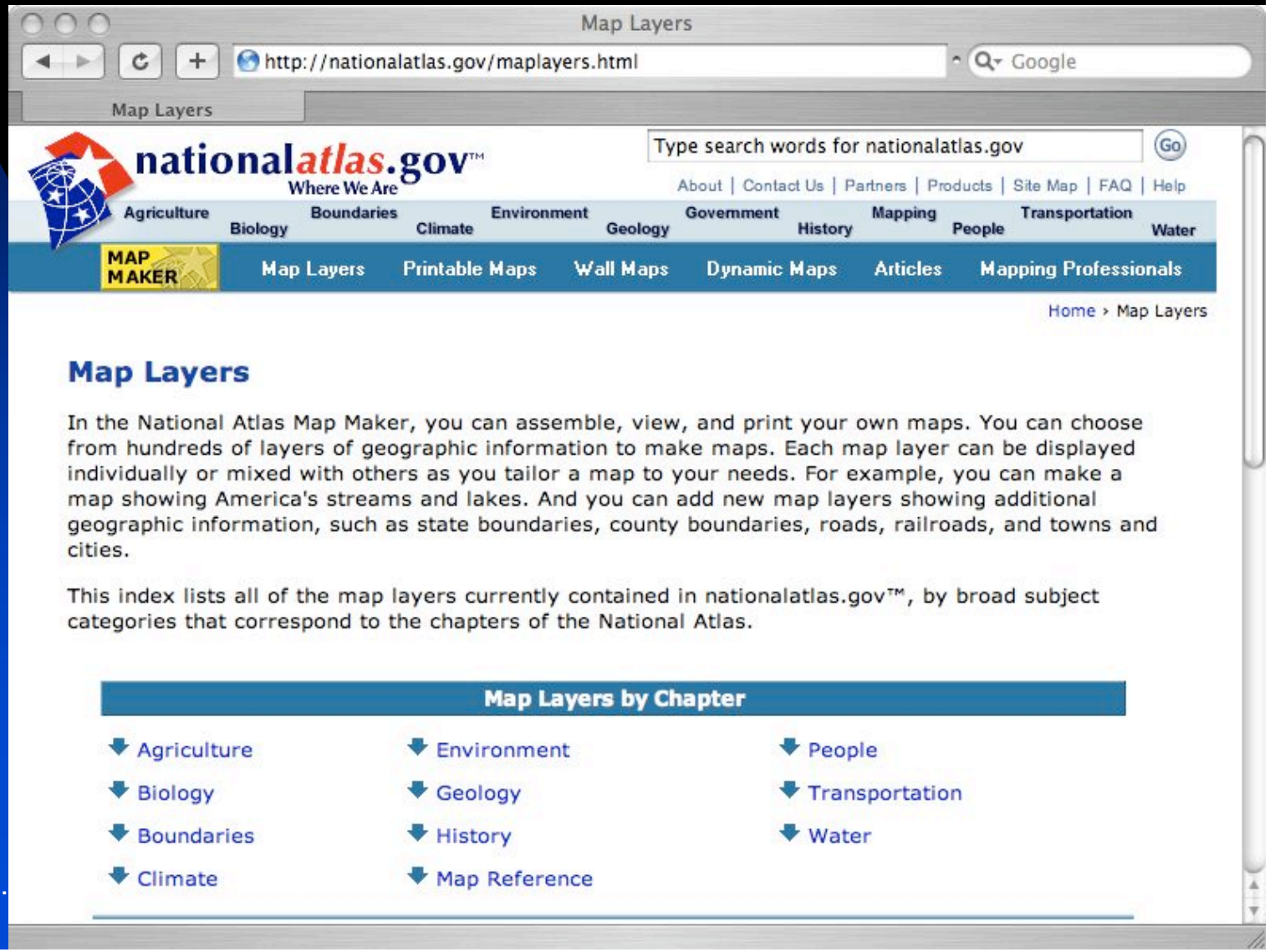
- Features can have non map-related data that comes along for the ride:
  - Right-click, Attribute Table

Data Layers

- You can add multiple layers to a map
  - Each layer can be styled



More Data, Please



More *International* Data, Please

- Canada:
  - http://www.geobase.ca/

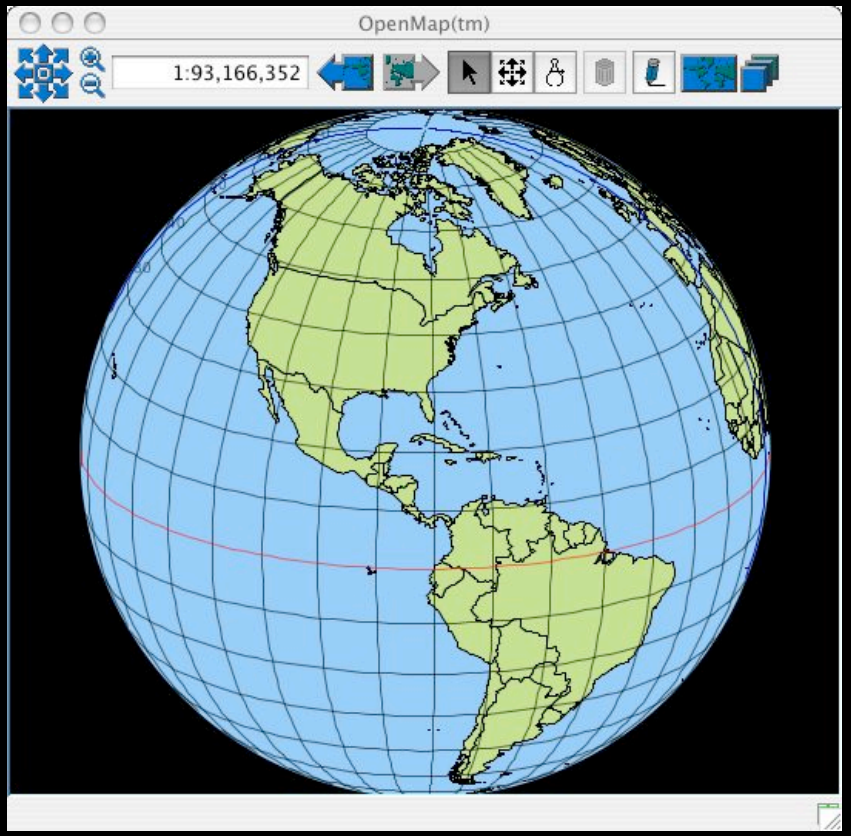
More *Local* Data, Please

- Most state and municipal government agencies offer free data as well

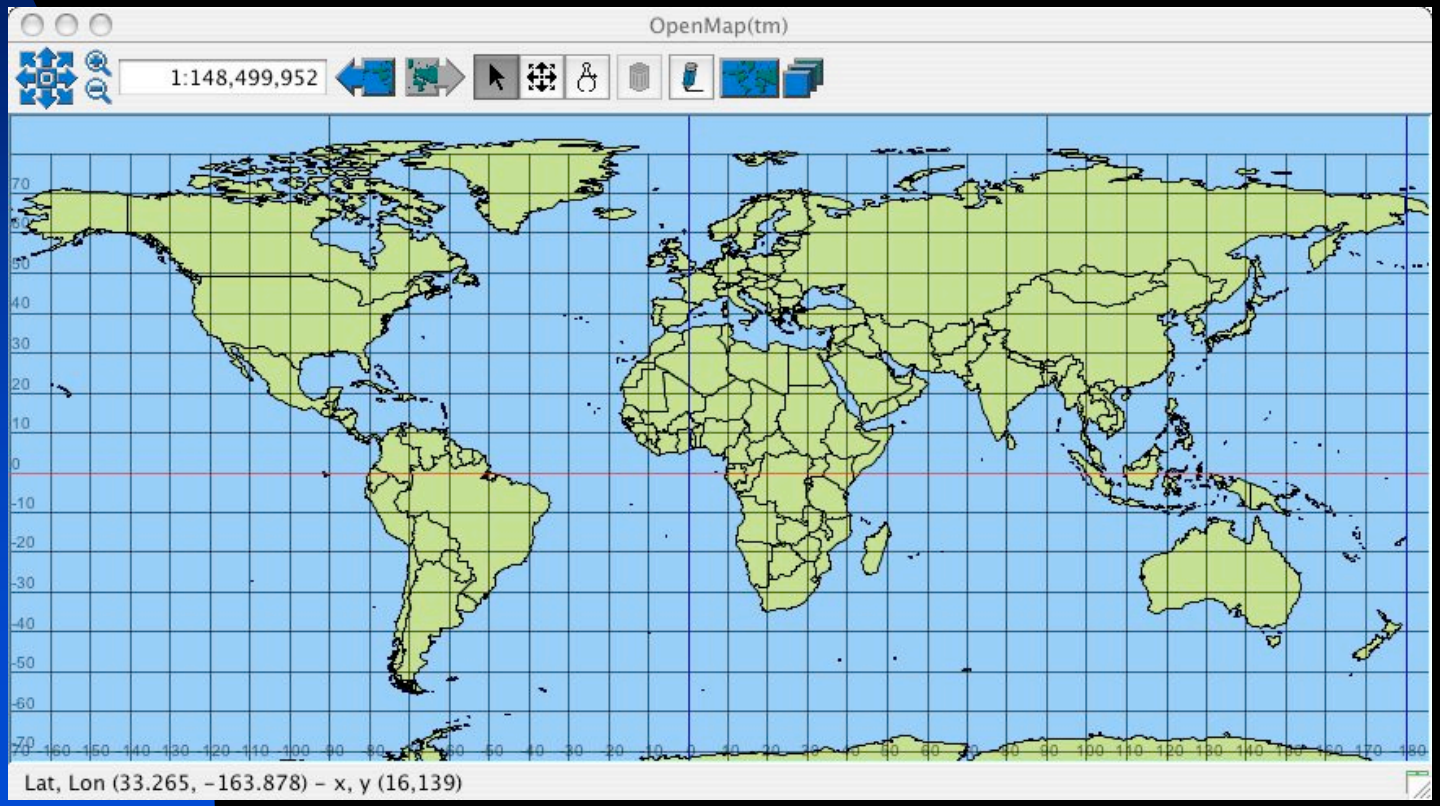
Act 2: Projections

When Good Data Goes Bad…

- Newsflash: The Earth is Round!



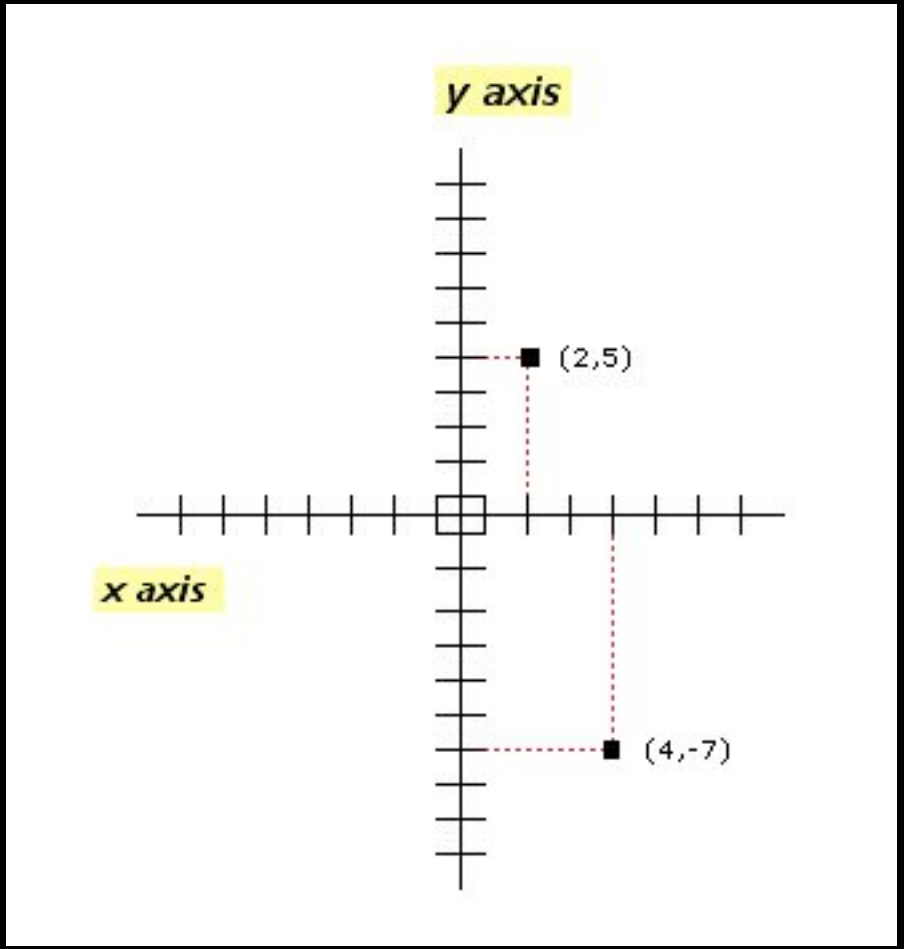
- This Just In: Maps are Flat!



Projections

- Map Projections:
  - Taking fundamentally 3-dimensional data and portraying it in 2-dimensions
  - This always introduces error:
    - Distance
    - Direction
    - Shape
    - Area

Cartesian Planes



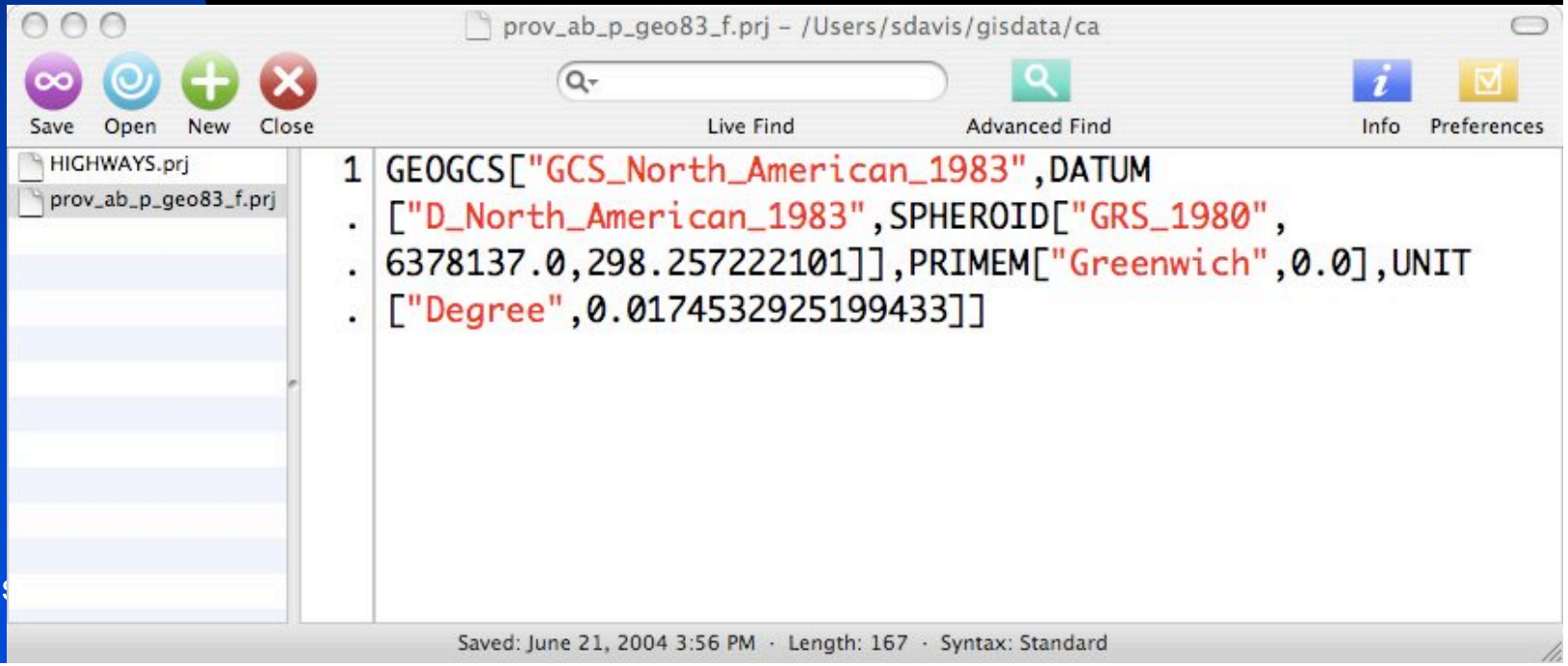
The (Mapping) Problem with Cartesian Planes

- On a Cartesian plane, the X and Y lines form perfect squares
- On a globe, only the lines of Latitude are perfectly parallel
  - Line of Longitude converge at the poles
  - This yields trapezoids that ultimately turn into triangles

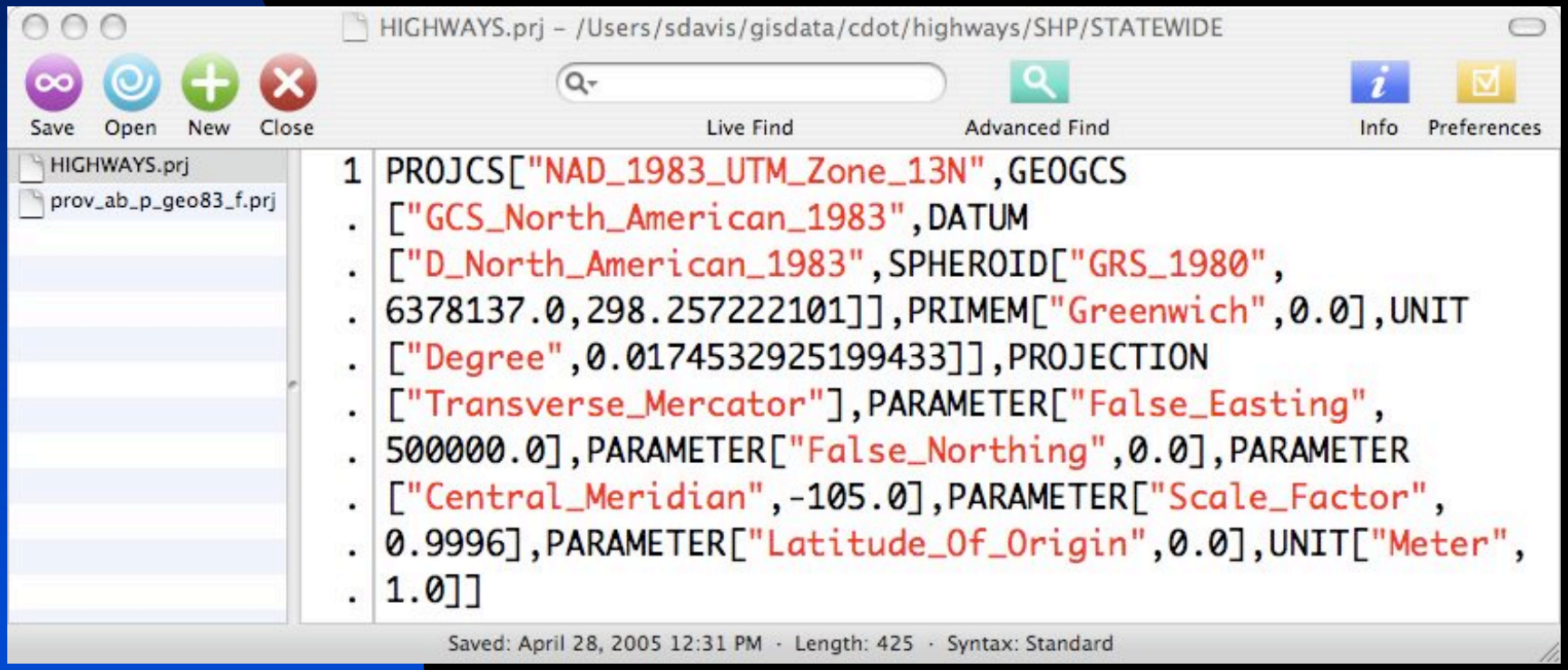
- Different projections attempt to minimize map distortions
  - Unprojected data is common at the World and Country level
  - Projected data is common at the State and Local level
    - Common projections are “State Plane” and “UTM”

Shapefile Projections

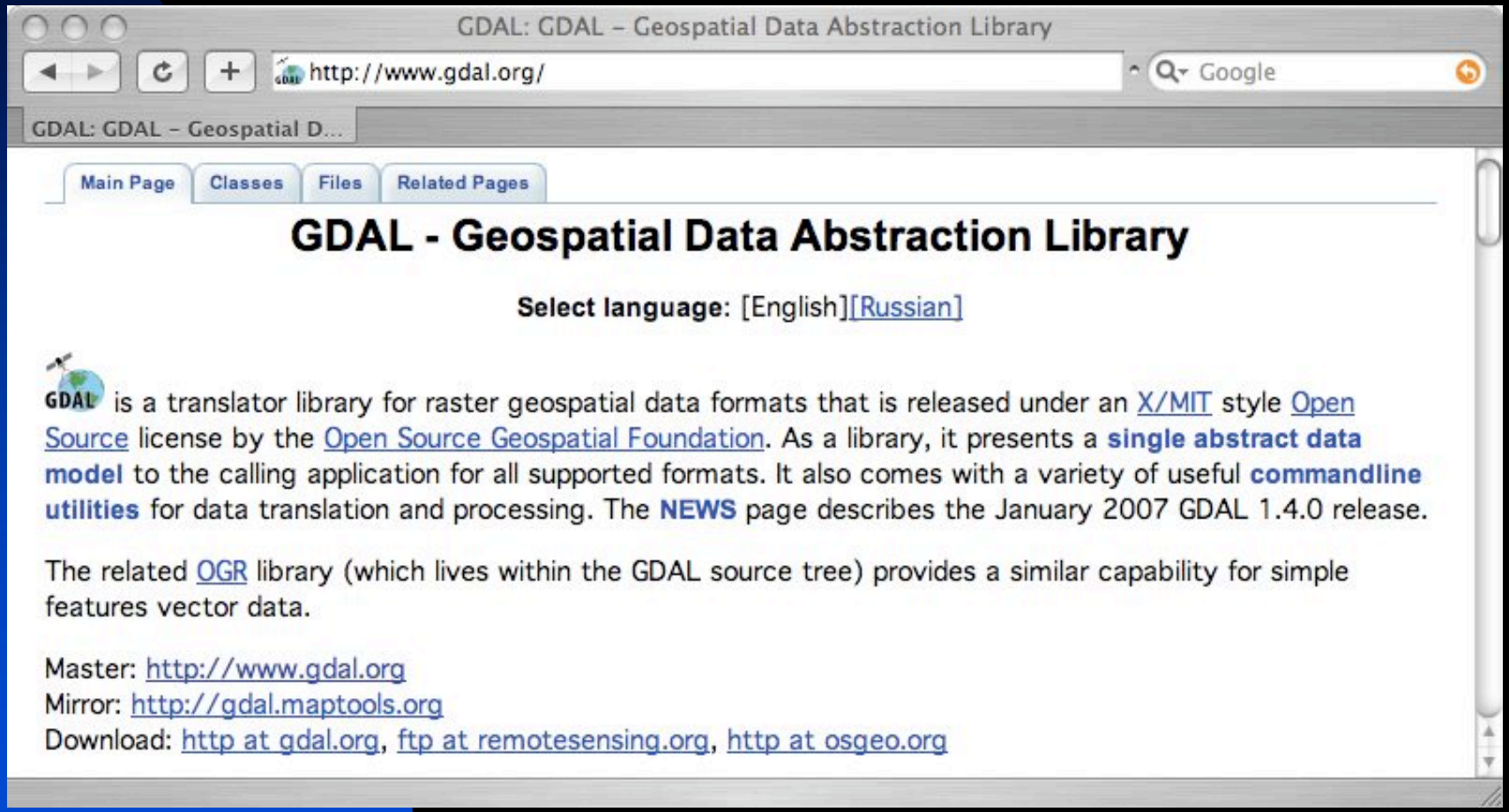
- If a shapefile has been reprojected, it should contain a .prj file
  - Canada:



- Colorado:

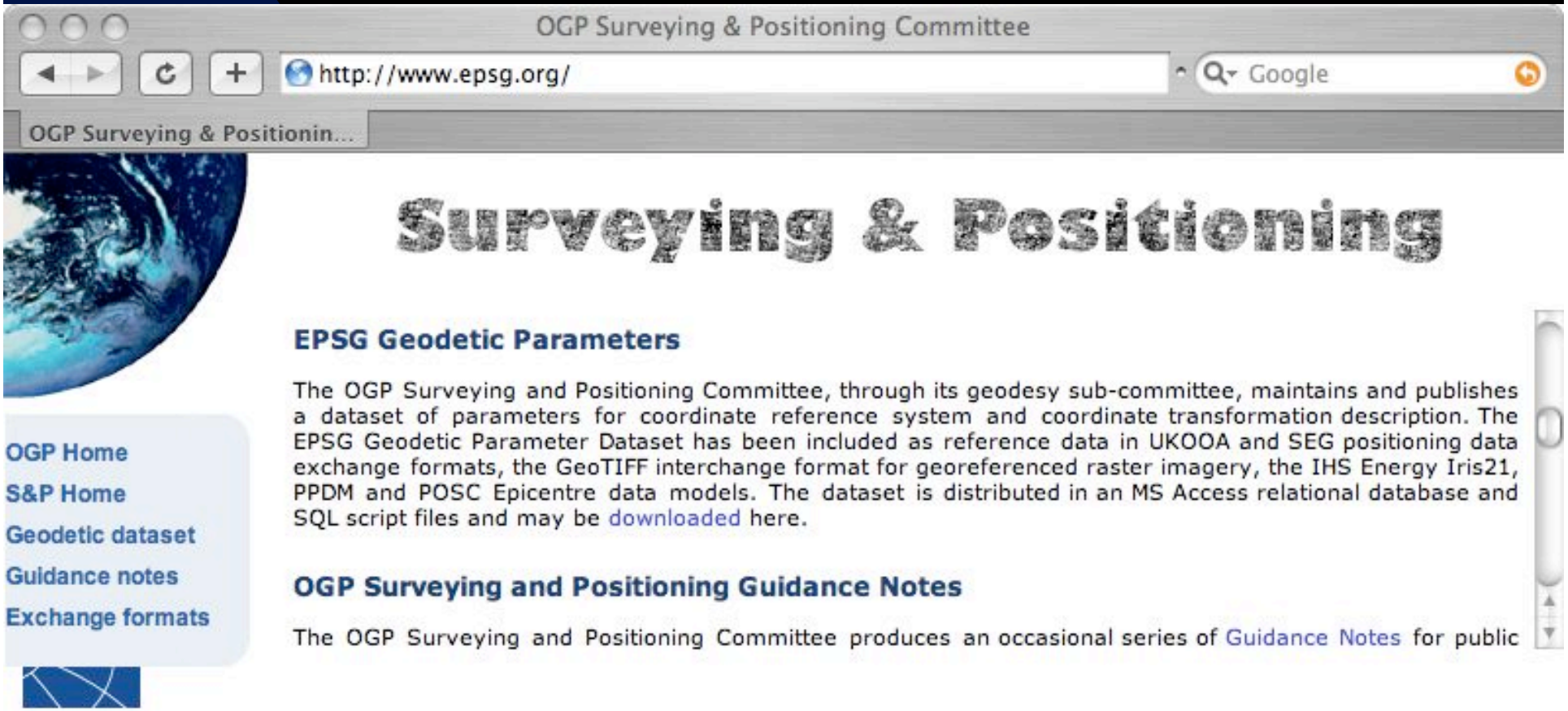


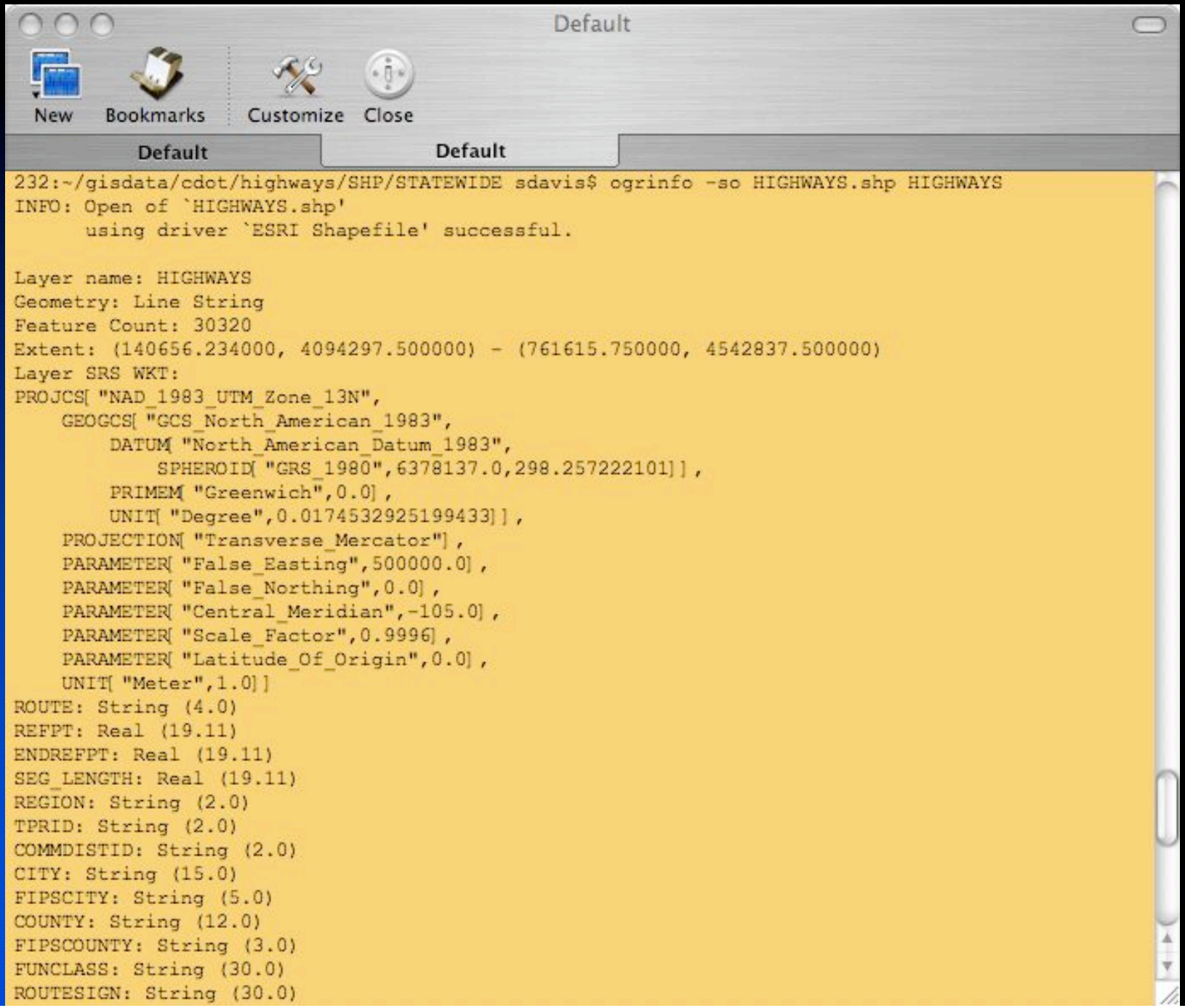
Reprojection Utility

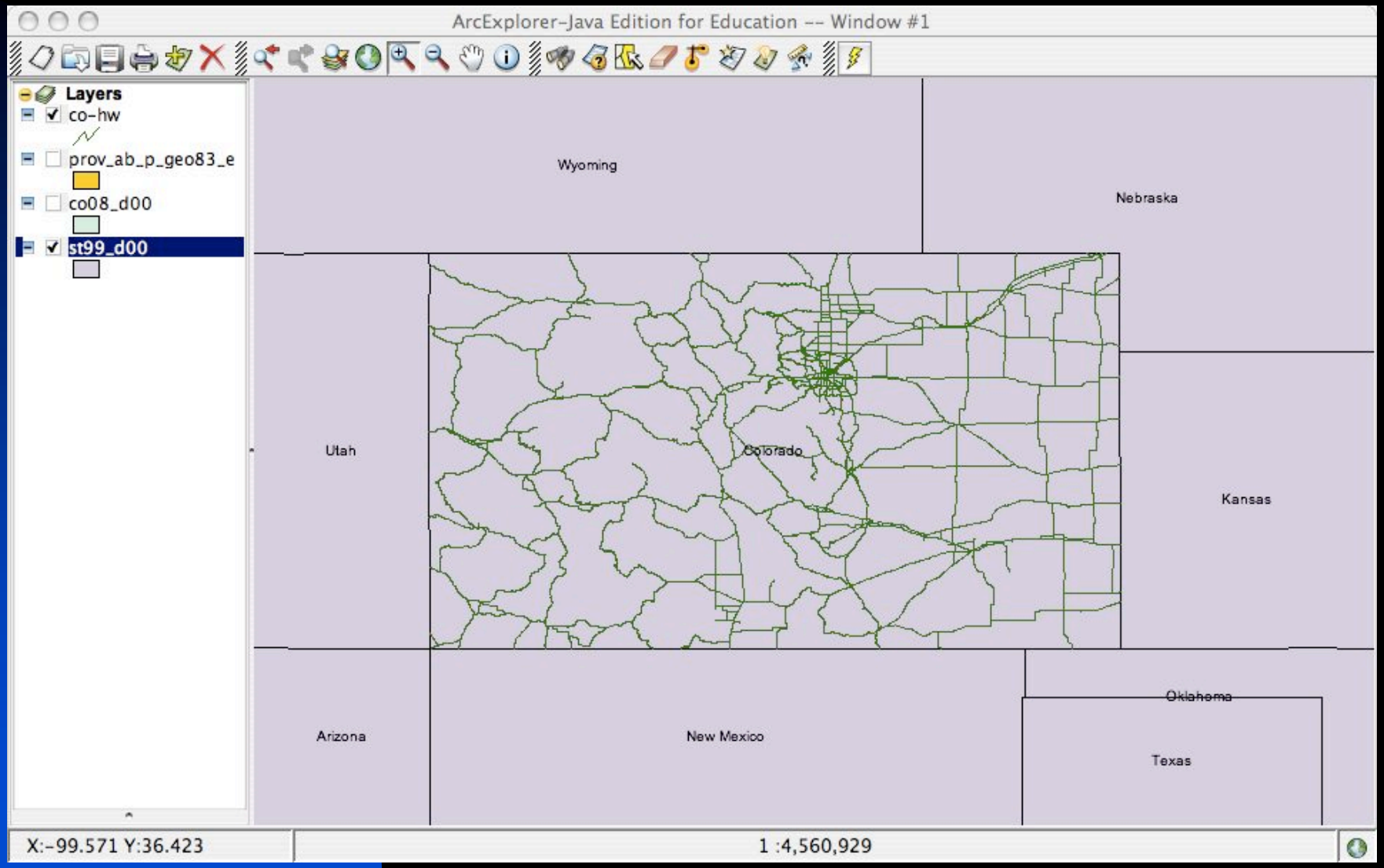


- To reproject from “UTM 13 N” to “Unprojected NAD83”
  - ogr2ogr -t_srs EPSG:4269 co-hw.shp highways.shp
- For info
  - ogrinfo -so highways.shp highways
- A very common projection is WGS 84, or EPSG: 4326

EPSG Codes



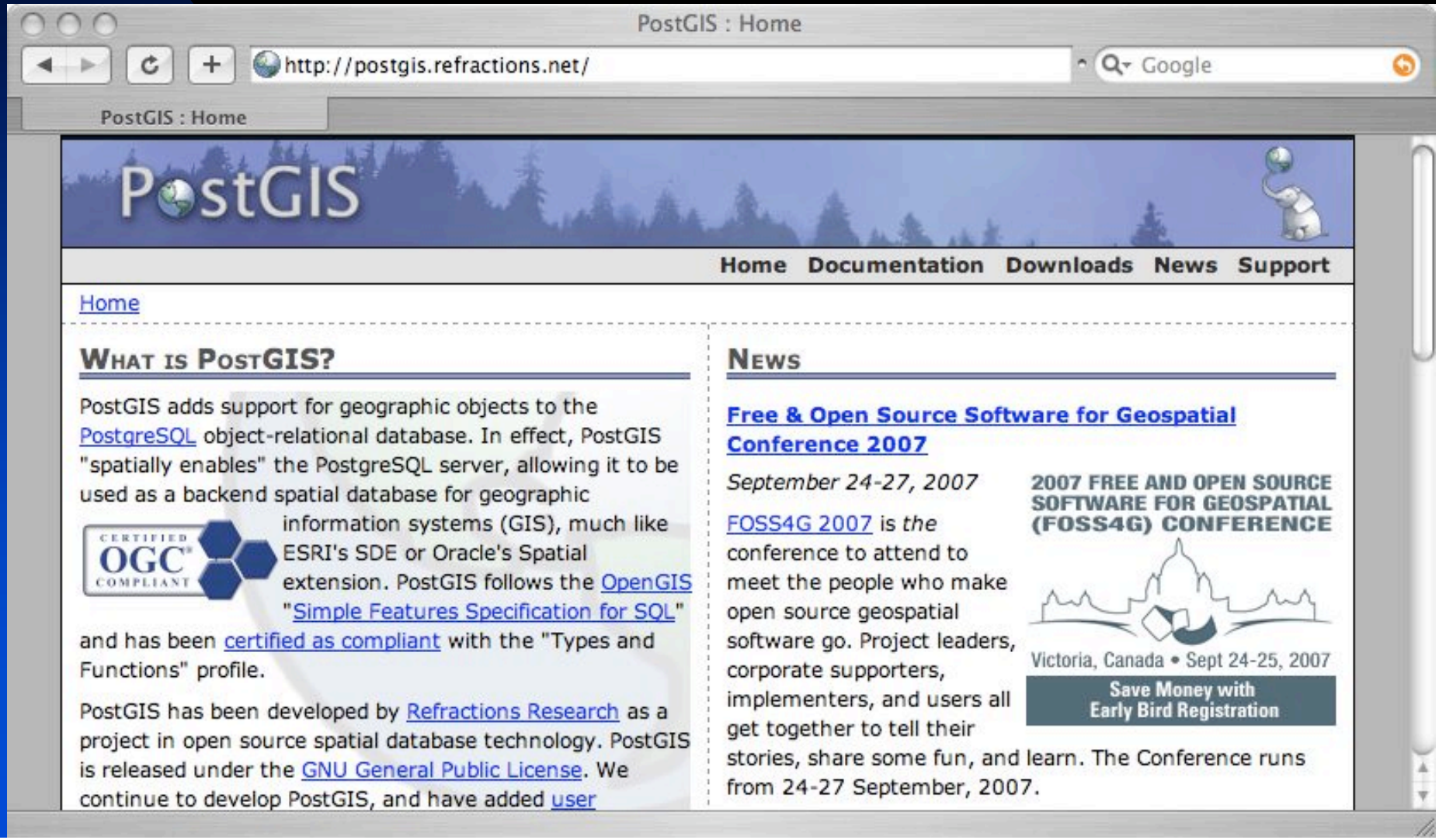

Default
New
Bookmarks
Customize
Close
Default
Default
232:~/gisdata/cdot/highways/SHP/STATEWIDE sdavis$ ogrinfo -so HIGHWAYS.shp HIGHWAYS
INFO: Open of `HIGHWAYS.shp'
using driver `ESRI Shapefile' successful.
Layer name: HIGHWAYS
Geometry: Line String
Feature Count: 30320
Extent: (140656.234000, 4094297.500000) - (761615.750000, 4542837.500000)
Layer SRS WKT:
PROJCS["NAD_1983_UTM_Zone_13N",
GEOGCS["GCS_North_American_1983",
DATUM["North_American_Datum_1983",
SPHEROID["GRS_1980",6378137.0,298.257222101]],
PRIMEM["Greenwich",0.0],
UNIT["Degree",0.0174532925199433]],
PROJECTION["Transverse_Mercator"],
PARAMETER["False_Easting",500000.0],
PARAMETER["False_Northing",0.0],
PARAMETER["Central_Meridian",-105.0],
PARAMETER["Scale_Factor",0.9996],
PARAMETER["Latitude_Of_Origin",0.0],
UNIT["Meter",1.0]]
ROUTE: String (4.0)
REFPT: Real (19.11)
ENDREFPT: Real (19.11)
SEG_LENGTH: Real (19.11)
REGION: String (2.0)
TPRID: String (2.0)
COMMDISTID: String (2.0)
CITY: String (15.0)
FIPSCITY: String (5.0)
COUNTY: String (12.0)
FIPSCOUNTY: String (3.0)
FUNCLASS: String (30.0)
ROUTESIGN: String (30.0)



Act 3:
Spatial Databases

- Why bother with a database?
  - Centralize many scattered files
  - Provide security
  - Indexing
  - Cross dataset queries

PostgreSQL + PostGIS

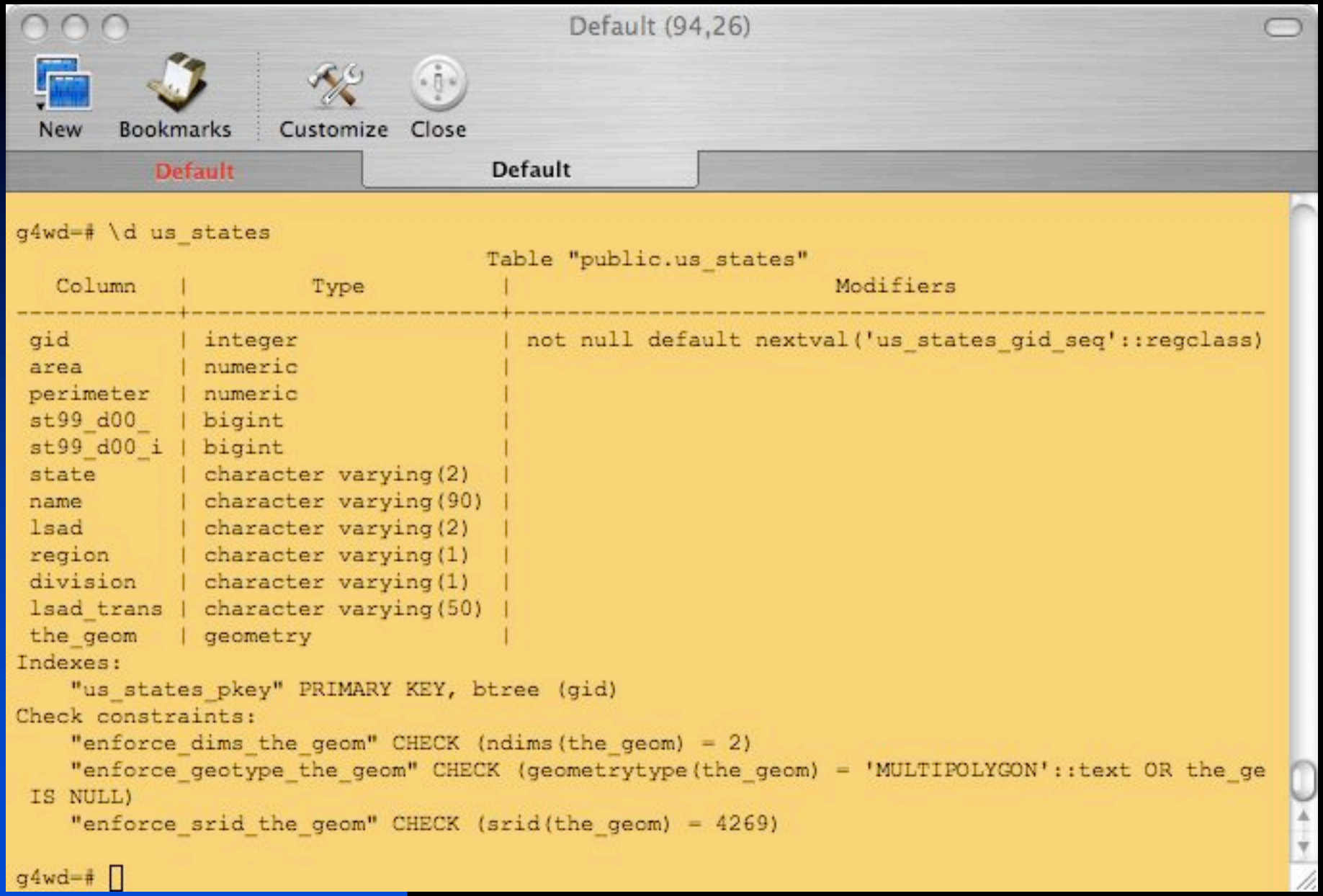


Importing Shapefiles

- Convert the Shapefile to a SQL import script
  - shp2sql -s 4269 st99_d00.shp us_states > us_states.sql
- Create the database
  - /usr/local/pgsql/bin/createdb g4wd
- Now import the SQL
  - psql -f us_states.sql -d g4wd

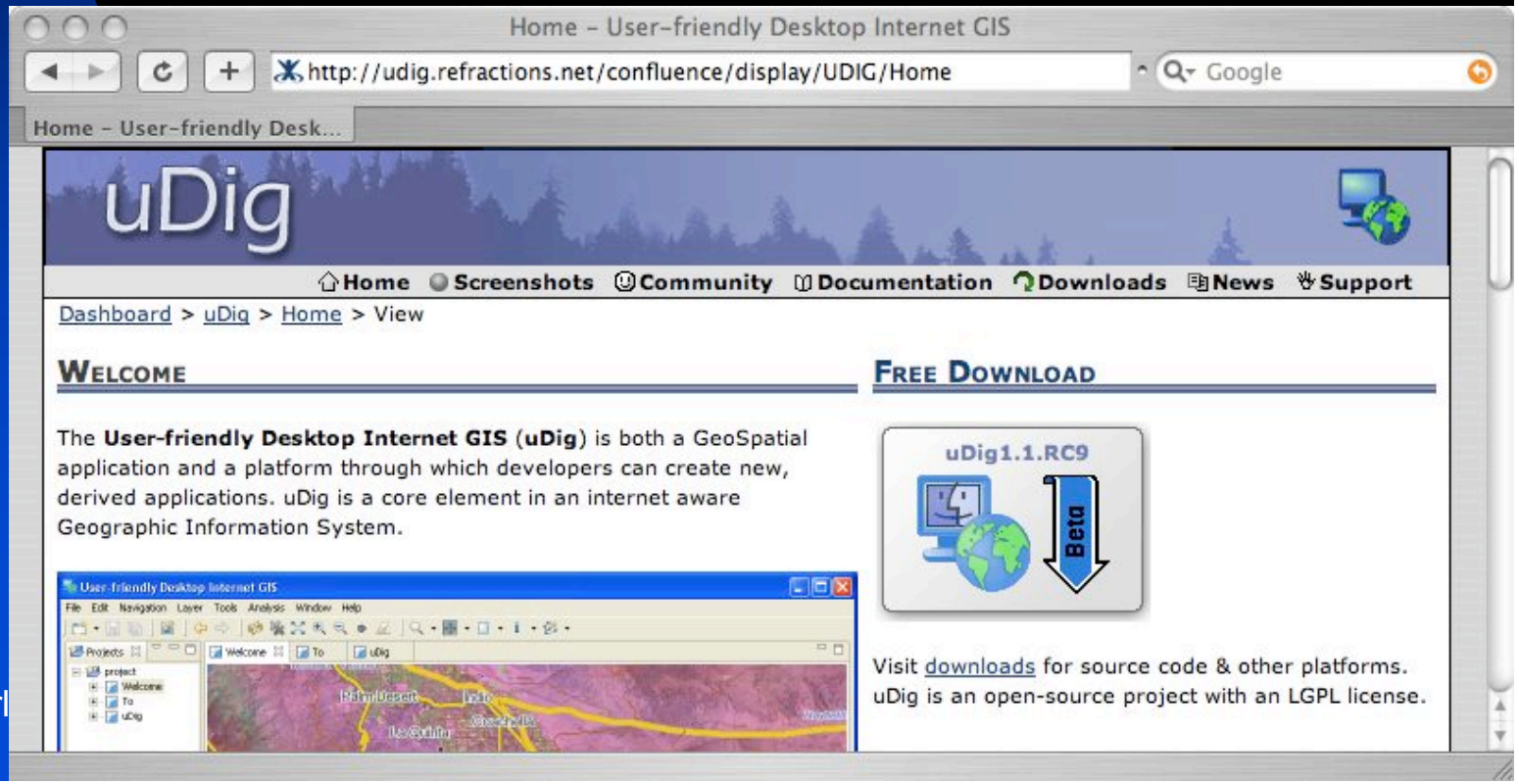
Looking Around

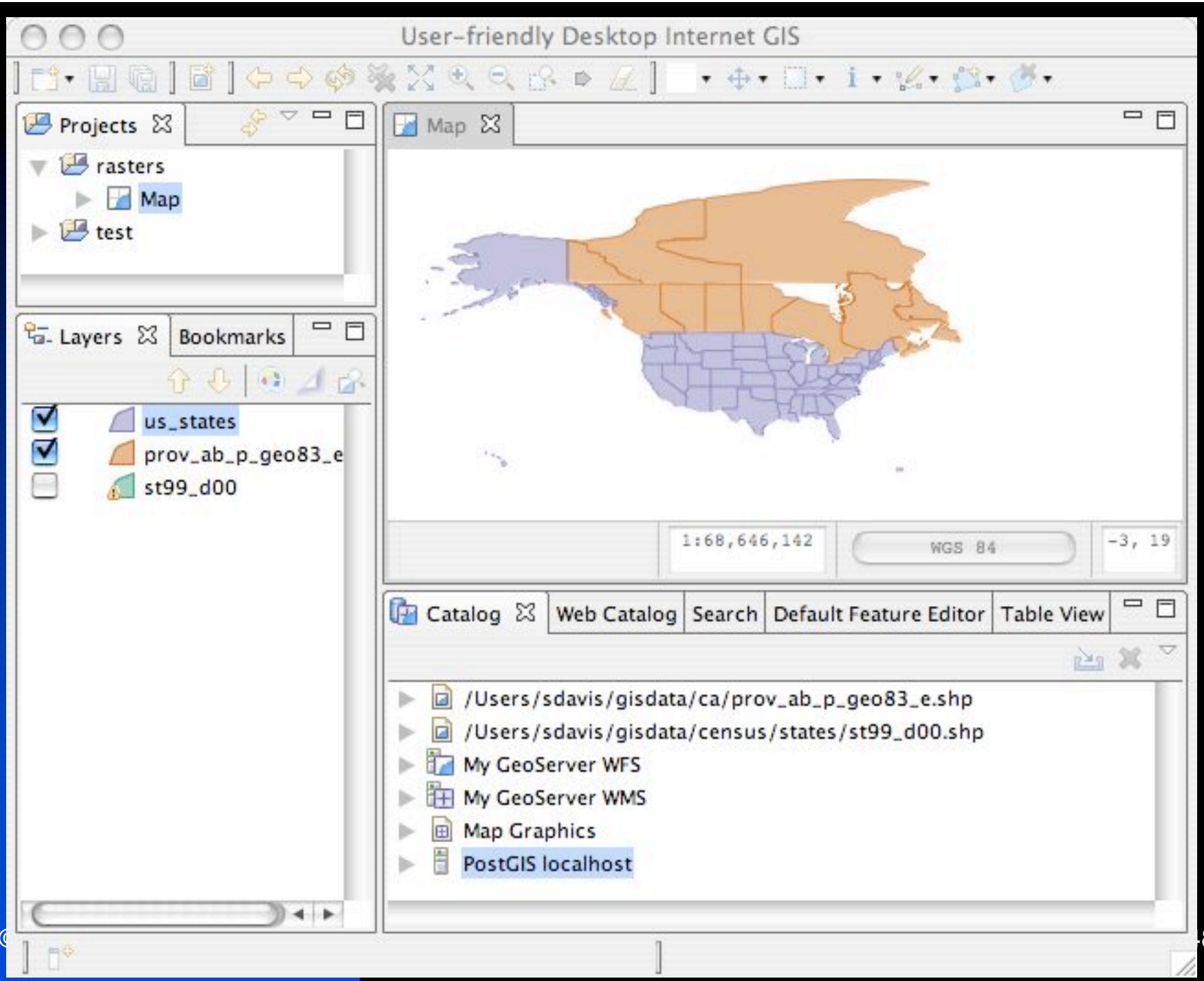
- Log into PostgreSQL
  - psql g4wd
- Look for the new table
  - \dt
- Describe the table schema
  - \d us_states



Visualizing the Data

- uDig
  - http://udig.refractions.net




User-friendly Desktop Internet GIS
Projects
rasters
Map
test
Layers
Bookmarks
us_states
prov_ab_p_geo83_e
st99_d00
Map
1:68,646,142
WGS 84
-3, 19
Catalog
Web Catalog
Search
Default Feature Editor
Table View
/Users/sdavis/gisdata/ca/prov_ab_p_geo83_e.shp
/Users/sdavis/gisdata/census/states/st99_d00.shp
My GeoServer WFS
My GeoServer WMS
Map Graphics
PostGIS localhost

Act 4:
Web Services

- Google Maps is undeniably cool, but it is a proprietary interface
- There is an open set of standards that allows you to do the same sort of thing…

Who is the OGC?

- The Open Geospatial Consortium
  - http://www.opengis.org

The Open Geospatial Consortium, Inc. (OGC) is a non-profit, international, voluntary consensus standards organization that is leading the development of standards for geospatial and location based services.

What does the OGC offer?

- Using the OGC interface, there are three basic things an application can ask:
  - What services do you offer?
  - What will the format of the output be?
  - Give me the data, please.

GetCapabilities

- The first thing a program will do is ask the OGC server what data layers it has to offer
  - This is called a GetCapabilities request

```
http://localhost:8888/geoserver/wfs?
service=WFS&
version=1.0.0&
request=GetCapabilities
```

GetCapabilities

- And here is the response:

```
- <FeatureTypeList>
  - <Operations>
      <Query/>
    </Operations>
  - <FeatureType>
      <Name>wfs:co_cities_poly</Name>
      <SRS>EPSG:4326</SRS>
      <LatLongBoundingBox minx="-109.018116" miny="37.00000485324387" maxx="-1
    </FeatureType>
  - <FeatureType>
      <Name>wfs:co_counties_poly</Name>
      <SRS>EPSG:4326</SRS>
      <LatLongBoundingBox minx="-180.0" miny="-90.0" maxx="180.0" maxy="90.0"/>
    </FeatureType>
  - <FeatureType>
      <Name>wfs:co_highways_line</Name>
      <SRS>EPSG:4326</SRS>
      <LatLongBoundingBox minx="-109.06008268030426" miny="36.9936887423584" n
    </FeatureType>
  - <FeatureType>
      <Name>wfs:us_states_poly</Name>
      <SRS>EPSG:4326</SRS>
      <LatLongBoundingBox minx="-179.14734" miny="17.884813" maxx="179.77847" r
    </FeatureType>
</FeatureTypeList>
```

DescribeFeatureType

- Once you find a layer that looks interesting, you can ask the service to describe it:

```
http://localhost:8888/geoserver/wfs?
version=1.0.0&
service=WFS&
request=DescribeFeatureType&
typename=us_states_poly
```

DescribeFeatureType

- And here is the response:

```
<xsd:schema targetNamespace="http://www.ionicsoft.com/wfs" elementFormDefault="qualified" version="0.1">
  <xsd:import namespace="http://www.opengis.net/gml" schemaLocation="http://www.opengis.net/namespaces/gml/core/feature.xsd"/>
  <xsd:element name="us_states_poly" substitutionGroup="gml:_Feature" type="wfs:us_states_poly"> </xsd:element>
- <xsd:complexType name="us_states_poly">
   - <xsd:complexContent>
      - <xsd:extension base="gml:AbstractFeatureType">
         - <xsd:sequence>
              <xsd:element name="AREA" minOccurs="0" nillable="true" type="xsd:double"> </xsd:element>
              <xsd:element name="PERIMETER" minOccurs="0" nillable="true" type="xsd:double"> </xsd:element>
              <xsd:element name="ST99_D00_" minOccurs="0" nillable="true" type="xsd:int"> </xsd:element>
              <xsd:element name="ST99_D00_I" minOccurs="0" nillable="true" type="xsd:int"> </xsd:element>
              <xsd:element name="STATE" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="NAME" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="LSAD" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="REGION" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="DIVISION" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="LSAD_TRANS" minOccurs="0" nillable="true" type="xsd:string"> </xsd:element>
              <xsd:element name="GEOMETRY" minOccurs="0" nillable="true" type="gml:PolygonPropertyType"> </xsd:element>
          </xsd:sequence>
       </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:schema>
```

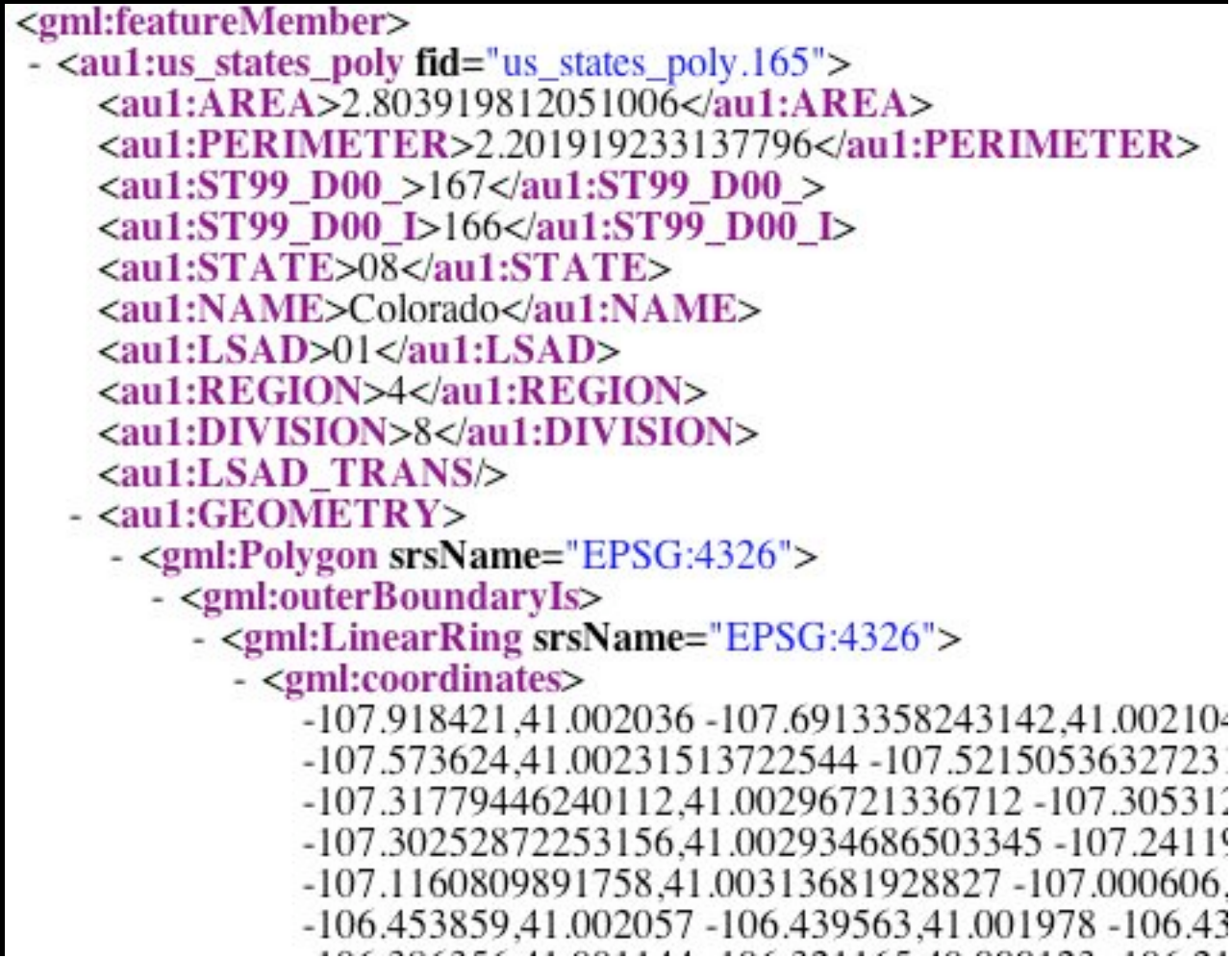
GetFeature

- Finally, you can request the data layer:

```
http://localhost:8888/geoserver/wfs?
version=1.0.0&
service=WFS&
request=GetFeature&
typename=us_states_poly
```

GetFeature

- And here is the response:
  - This is called GML (Geometry Markup Language)



(…Astoundingly boring, isn't it?)

Data vs Pretty Pixels

- Data services:
  - WFS (Web Feature Service)
- Portrayal services:
  - WMS (Web Mapping Service)

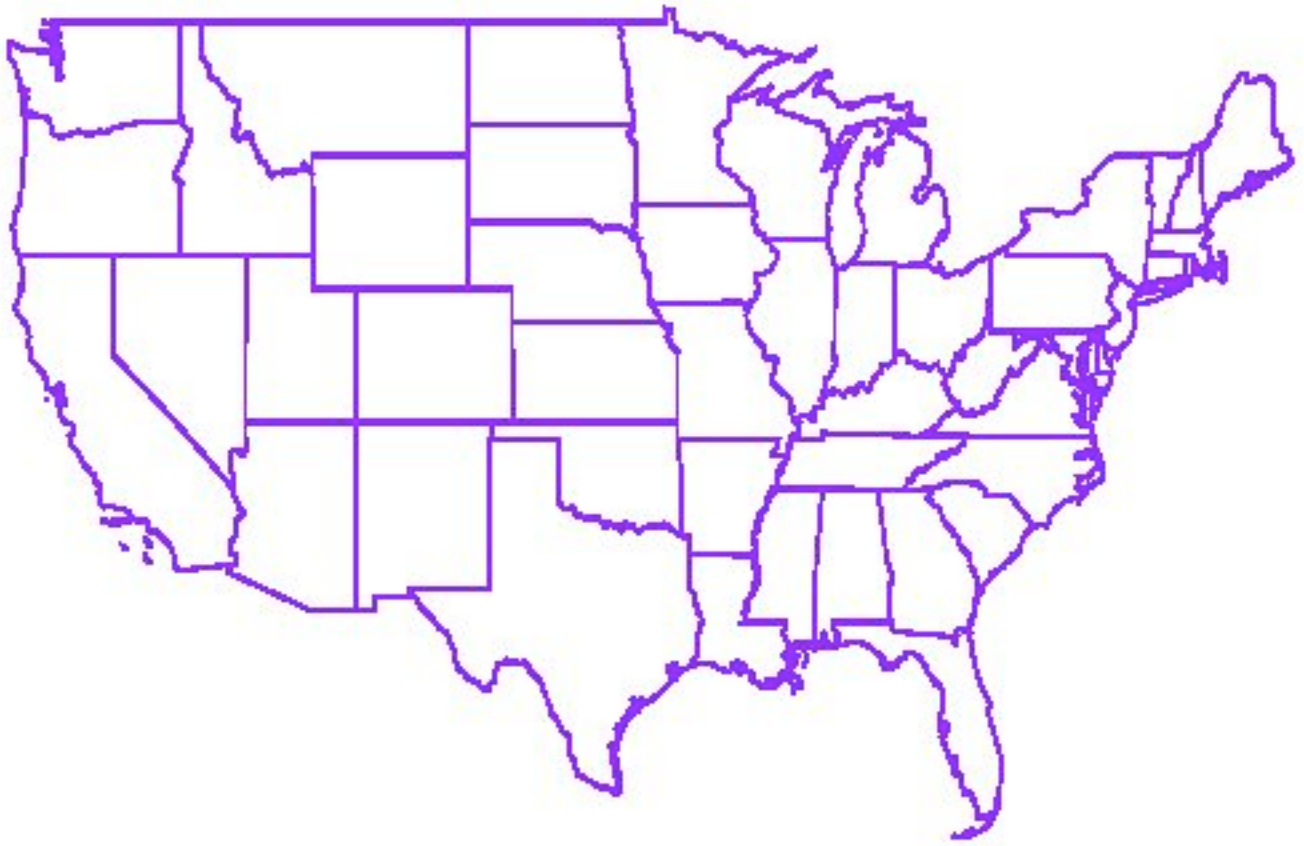
Web Mapping Service

- GetMap request:

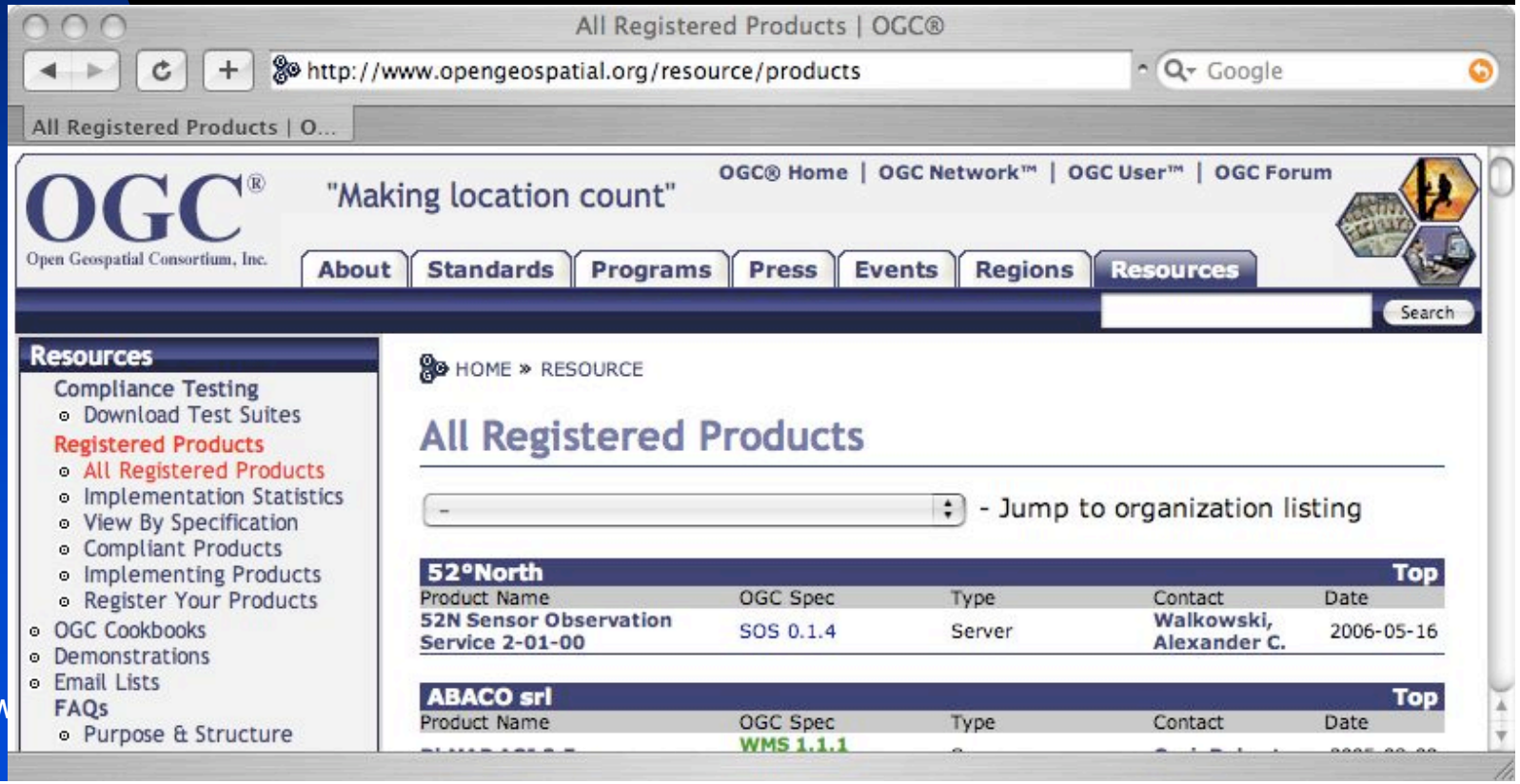
```
http://localhost:8888/geoserver/wms?
VERSION=1.1.1&
REQUEST=GetMap&
SRS=EPSG:4326&
BBOX=-126,20,-66,52&
WIDTH=500&
HEIGHT=500&
LAYERS=us_states_poly&
STYLES=&
FORMAT=image/png&
BGCOLOR=0xffffff&
TRANSPARENT=FALSE&
EXCEPTIONS=application/vnd.ogc.se_inimage
```

Web Mapping Service

- …returns a single layer

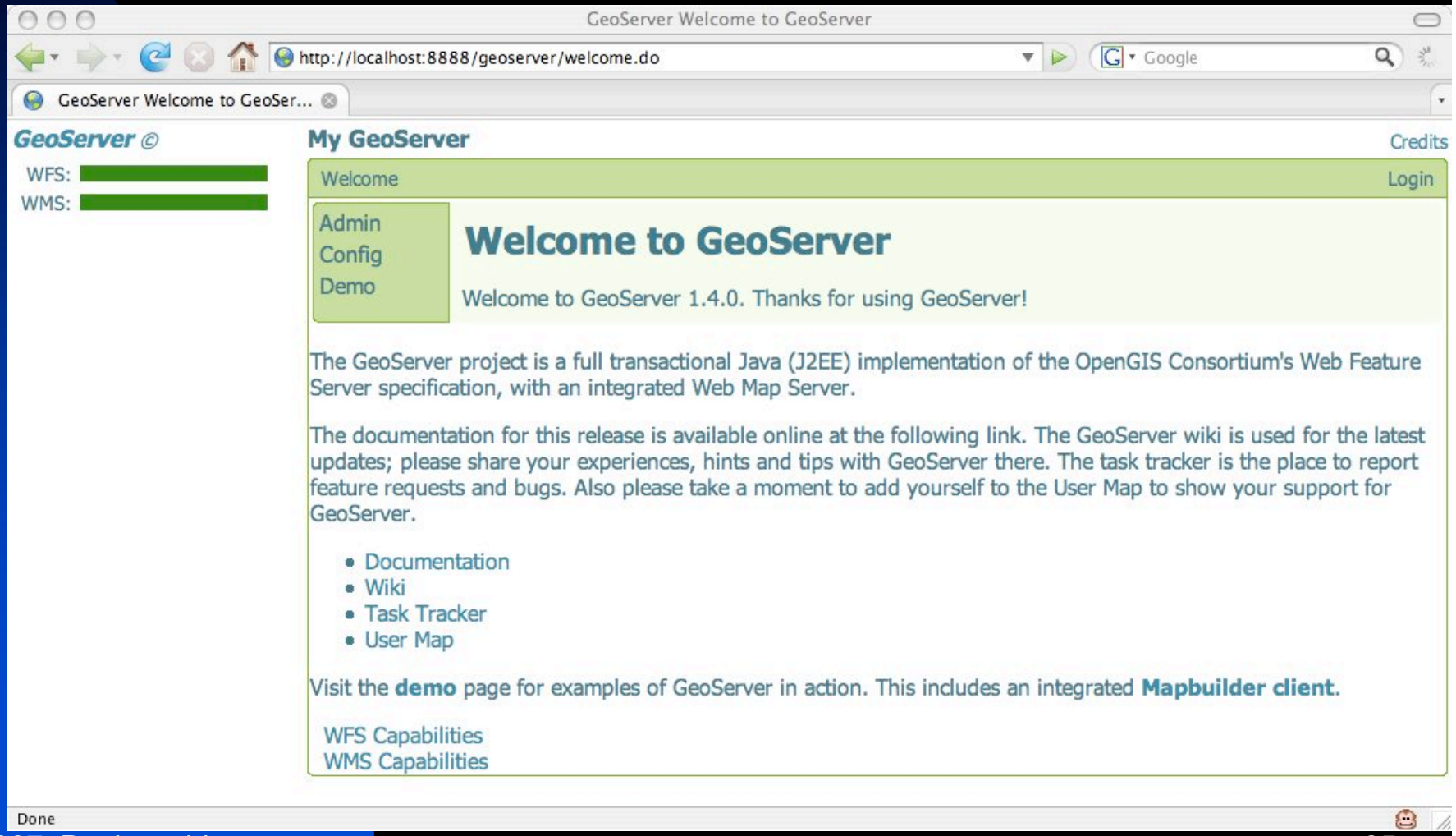


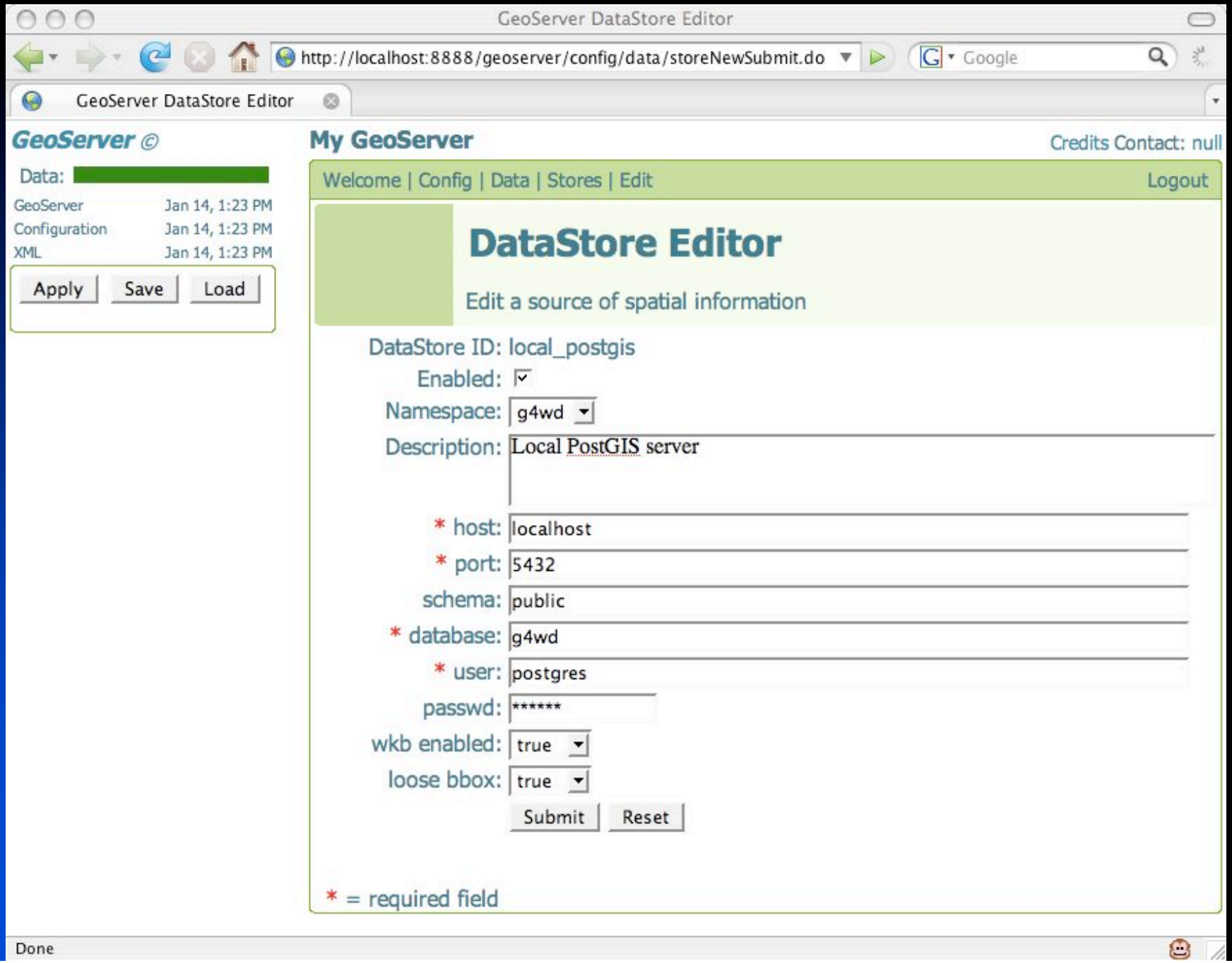
- Since the OGC standards are open, anyone can implement them:



- Java-based
  - http://docs.codehaus.org/display/GEOS/Home
- Runs in a servlet container
- Provides OGC (Open Geospatial Consortium) web services
  - WMS
  - WFS

Welcome Screen




GeoServer DataStore Editor
http://localhost:8888/geoserver/config/data/storeNewSubmit.do
Google
GeoServer DataStore Editor
GeoServer ©
Data:
GeoServer Jan 14, 1:23 PM
Configuration Jan 14, 1:23 PM
XML Jan 14, 1:23 PM
Apply
Save
Load
My GeoServer
Credits Contact: null
Welcome | Config | Data | Stores | Edit
Logout
DataStore Editor
Edit a source of spatial information
DataStore ID: local_postgis
Enabled:
Namespace: g4wd
Description: Local PostGIS server
* host: localhost
* port: 5432
schema: public
* database: g4wd
* user: postgres
passwd: ******
wkb enabled: true
loose bbox: true
Submit
Reset
* = required field
Done

MapBuilder

- Pure JavaScript web client for OGC
  - http://mapbuilder.sourceforge.net
- Included with GeoServer

Mix ‘n Match

- Since OGC is a standard interface, you can mix data from across the web with your own local layers

NASA + Live Weather

```
<LayerList>
  <Layer queryable="1" hidden="0">
    <Server service="OGC:WMS" version="1.1.1" title="blue marble">
      <OnlineResource xlink:type="simple" xlink:href="http://wms.jpl.nasa.gov/wms.cgi?"/>
    </Server>
    <Name>BMNG</Name>
    <Title>Blue Marble</Title>
    <SRS>EPSG:4326</SRS>
    <FormatList><Format current="1">image/png</Format></FormatList>
  </Layer>
  <Layer queryable="1" hidden="0">
    <Server service="OGC:WMS" version="1.1.1" title="weather">
      <OnlineResource xlink:type="simple" xlink:href="http://mesonet.agron.iastate.edu/cgi-bin/wms/nexrad/n0r.cgi"/>
    </Server>
    <Name>nexrad-n0r-m45m</Name>
    <Title>Weather</Title>
    <SRS>EPSG:4326</SRS>
    <FormatList><Format current="1">image/png</Format></FormatList>
  </Layer>
  <Layer queryable="1" hidden="0">
    <Server service="OGC:WMS" version="1.1.1" title="g4wd:st99_d00 Preview">
      <OnlineResource xlink:type="simple" xlink:href="../wms"/>
    </Server>
    <Name>g4wd:st99_d00</Name>
    <Title>US</Title>
    <SRS>EPSG:4326</SRS>
    <FormatList><Format current="1">image/png</Format></FormatList>
  </Layer>
```

g4wd:st99_d00 Preview

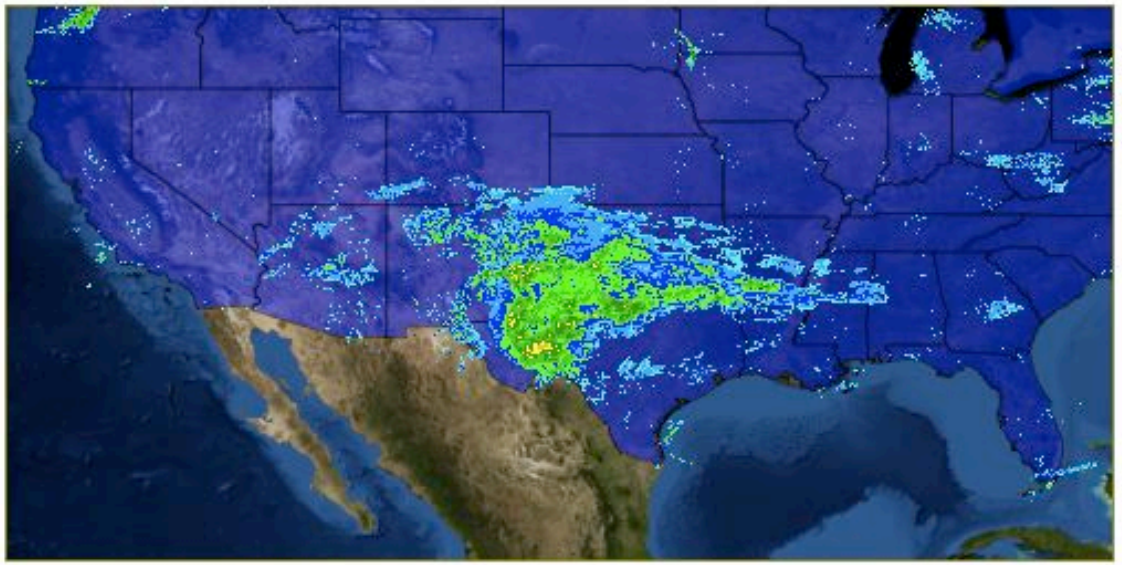
http://localhost:8888/geoserver/g4wd/g4wd_st99_d00.html

Google My Yahoo! Test Wikipedia Google Calendar Davisworld.org wiki Tomcat JAW Motors ryogm Guerrilla MapMap

OnEarth, server of the Landsat... | g4wd:st99_d00 Preview

GeoServer/Map Builder g4wd:st99_d00 preview

lon: lat:

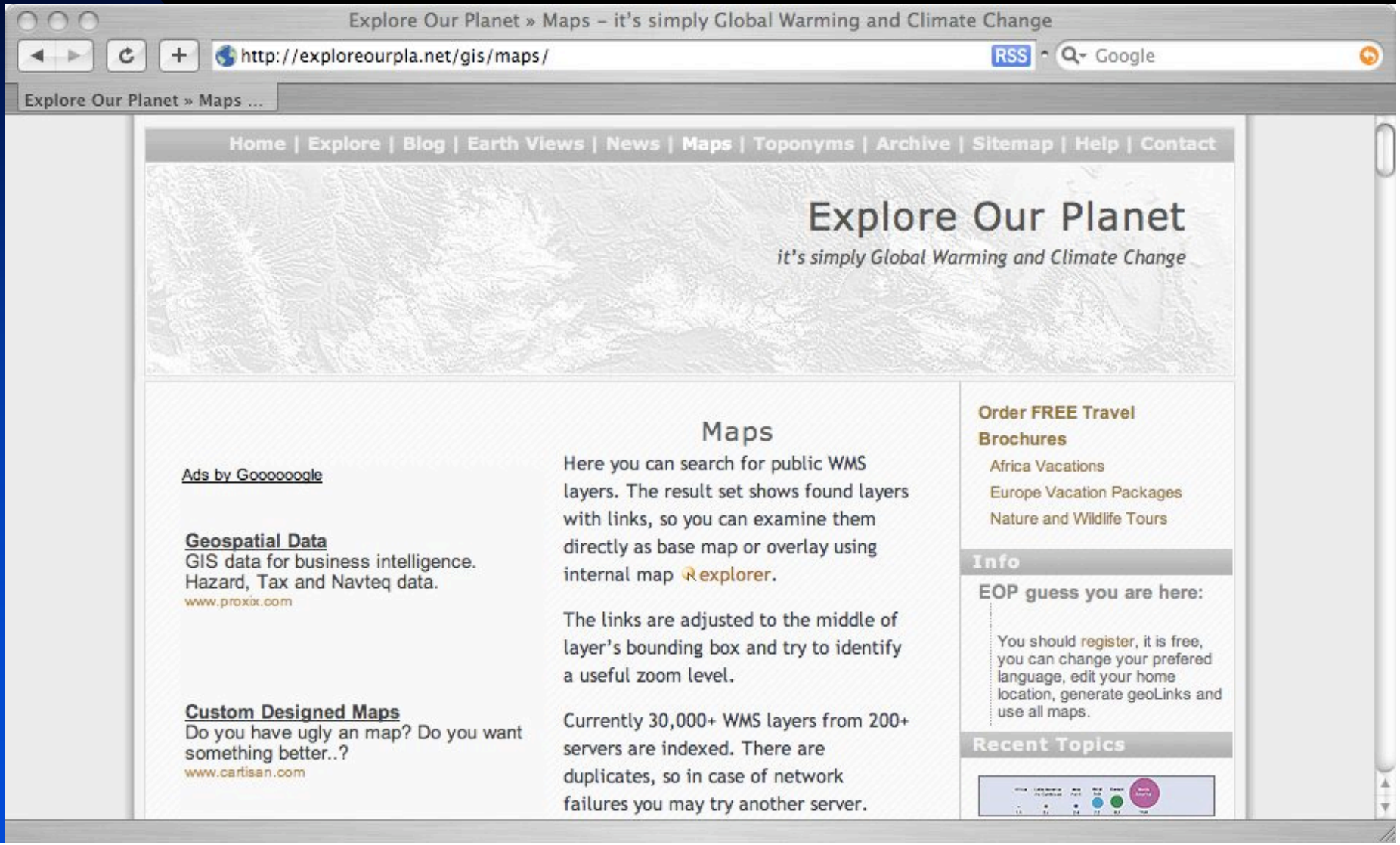


- Weather
- Canada
- US
- Blue Marble

scale 1:37,702,570 Powered by Community Map Builder

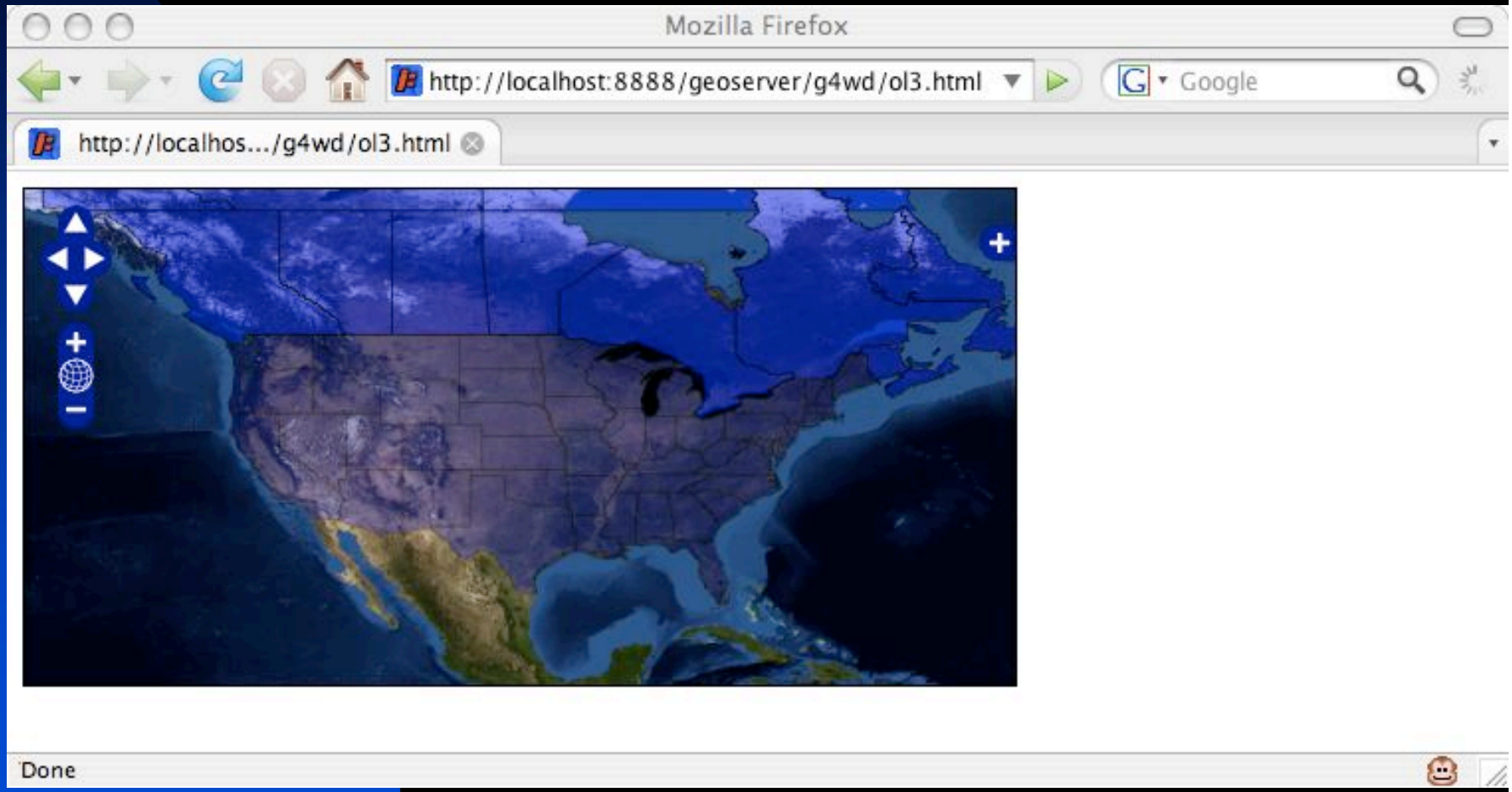
Done

ExploreOurPla.net



- Pure JavaScript web client
  - http://www.openlayers.org
- In addition to supporting OGC layers
  - Google Maps
  - Yahoo Maps
  - MS Virtual Earth (Local Live)
  - WorldWind

One Map…



…One File

```
10      <script src="http://www.openlayers.org/api/OpenLayers.js"></script>
11      <script type="text/javascript">
12          //NOTE: geographic center of the US
13          var lon = -98.583333;
14          var lat = 39.833333;
15          var zoom = 3;
16          var map, us, canada, blueMarble;
17
18          function init(){
19              map = new OpenLayers.Map( $('map') );
20              blueMarble = new OpenLayers.Layer.WMS( "Blue Marble",
    "http://wms.jpl.nasa.gov/wms.cgi?", {layers: 'BMNG', format: 'image/png'},
    {isBaseLayer:true});
21              map.addLayer(blueMarble);
22
23              us = new OpenLayers.Layer.WMS( "US", "http://localhost:8888/geoserver/wms?",
    {layers: 'g4wd:st99_d00', format: 'image/png', transparent: true}, {isBaseLayer:false,
    opacity:0.5} );
24              map.addLayer(us);
25
26              canada = new OpenLayers.Layer.WMS( "Canada",
    "http://localhost:8888/geoserver/wms?", {layers: 'g4wd:prov_ab_p_geo83_e', format:
    'image/png', transparent: true}, {isBaseLayer:false, opacity:1.0} );
27              map.addLayer(canada);
28
29              map.setCenter(new OpenLayers.LonLat(lon, lat), zoom);
30              map.addControl( new OpenLayers.Control.LayerSwitcher() );
31          }
32      </script>
```

Conclusion

- You now know:
  - Vector vs. Raster data
    - Shapefiles
    - Free sources of data and viewers
  - Projections
    - GDAL and OGR2OGR
  - Spatial Databases
    - PostgreSQL + PostGIS
  - Web Services
    - OGC
    - GeoServer
    - MapBuilder
    - OpenLayers

Conclusion

- Thanks for your time!
  - Questions?
  - Email:
    - scottdavis99@yahoo.com
  - Download slides:
    - http://www.davisworld.org/presentations