

# Java Topology Suite in Action

---

Combining ESRI and Open Source



Jared Erickson  
Pierce County, WA

# Introduction

---

- ▣ Combing ESRI and Open Source GIS
- ▣ The Java Topology Suite (JTS)
- ▣ How Pierce County uses it with ESRI software
- ▣ How Pierce County extends it

# ESRI and Open Source

---



# What is JTS?

---

- Java API for Vector Geometry
  - Geometry Object Model
  - Geometry functions, Spatial Predicates, Overlay Methods, and Algorithms
- Open source
- Written by Martin Davis of Refractions Research
- Implements OGC Simple Features for SQL specification (no curves)

# What is it?

---

- Core library in Java tribe of open source GIS
- JTS code is ported to C/C++ as GEOS
  - GEOS is the core library of the C open source tribe
  - GEOS is used by PostGIS, GDAL/OGR, MapServer, QGIS, Shapely (Python)
- Ported to .NET as NetTopologySuite
- Explicit Precision Model
- Focuses on Robustness vs. Speed

# Geometry

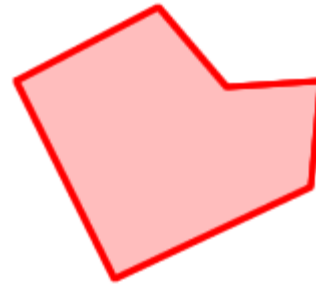
---



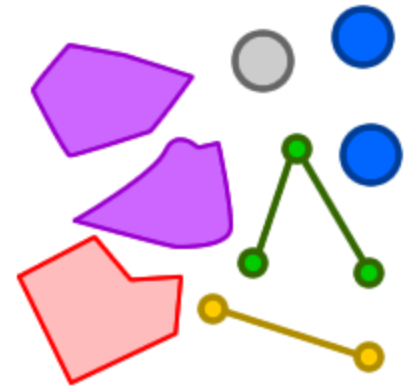
**Point**



**LineString**



**Polygon**



**GeometryCollection**



**MultiPoint**



**MultiLineString**



**MultiPolygon**

# Geometry Code Sample

---

```
GeometryFactory geometryFactory = new GeometryFactory();
```

```
Point point = geometryFactory.createPoint(new  
    Coordinate(200.0, 323.0));
```

```
LineString lineString = geometryFactory.createLineString(new  
    Coordinate[] {  
        new Coordinate(2.2,3.3),  
        new Coordinate(4.4,5.5),  
        new Coordinate(6.6,7.7)  
    });
```

# Spatial Functions and Predicates

---

- ❑ Buffer
- ❑ Contains
- ❑ ConvexHull
- ❑ CoveredBy
- ❑ Covers
- ❑ Crosses
- ❑ Difference
- ❑ Disjoint
- ❑ Distance
- ❑ Equals
- ❑ Area
- ❑ Boundary
- ❑ Centroid
- ❑ Envelope
- ❑ EnvelopeInternal
- ❑ Length
- ❑ Intersection
- ❑ Intersects
- ❑ Is Empty
- ❑ Is Simple
- ❑ Is Valid
- ❑ Is Within Distance
- ❑ Normalize
- ❑ Overlaps
- ❑ Relate (DE-9IM Intersection Matrix)
- ❑ SymDifference
- ❑ Touches
- ❑ Union
- ❑ Within

# Spatial Functions and Predicates

---

```
GeometryFactory geometryFactory = new GeometryFactory();
```

```
Point point = geometryFactory.createPoint(new  
    Coordinate(200.0, 323.0));
```

```
Geometry bufferedPoint = point.buffer(1000.00);
```

```
LineString lineString = geometryFactory.createLineString(new  
    Coordinate[] {  
        new Coordinate(2.2,3.3),  
        new Coordinate(4.4,5.5),  
        new Coordinate(6.6,7.7)  
    });
```

```
Geometry envelope = lineString.getEnvelope();
```

# Algorithms

---

- Validation
- Line Merging
- Polygonization
- Spatial Indexes (Quad Tree, STRtree, BinTree...)
- Linear Referencing
- Planar graphs
- Simplification (Douglas Peucker, Topology Preserving)

# Who uses it?

---



uDig  
User-friendly Desktop Internet GIS



# JTS usage in Pierce County

---

- CountyView Web
- Pierce County GIS Web Services
- JTSIO
- JTS Web Processing



# CountyView Web

---

- Entry Level Enterprise GIS
- Built on IMF, ArcIMS, ArcSDE
- Data Menu, Locator, Owner Notify, Metadata, Census, Printing, Open @
- Focus:
  - Data Viewer
  - Printing
  - Integration
  - Ease of use

# CountyView Web

---

- All geometry in AXL requests is translated to JTS geometry by the IMF framework
- Owner Notify
- Reverse Geocode
- Profile

# CountyView Web: Owner Notify

The image shows a computer screen with two windows. The left window is a Mozilla Firefox browser displaying the CountyView Web interface. The browser's address bar shows the URL <http://shasta405.co.pierce.wa.us/imf/imf.jsp?site=prc>. The page title is "CountyView Web - Mozilla Firefox". The main content area is titled "Owner Notify" and displays the message: "You have 38 Tax Parcel(s) Selected! (TwoLotsDeep DeliveryAddress)". Below this message, there are several "Zoom to Parcel" buttons, each with a magnifying glass icon. The right window is an Adobe Reader application displaying a PDF document titled "[extractToMailingLabel.pdf]". The PDF content is a list of mailing labels, organized into columns. The browser's status bar at the bottom shows "Scale: 1:1,931" and "Map Tool: Select by Box".

# CountyView Web: Owner Notify

---

- ❑ Buffer the user's selected map feature:

```
Geometry bufferedGeometry = geometry.buffer(bufferDistance);
```

- ❑ Use the buffered geometry to perform a spatial query to return parcels
- ❑ If using lots deep option, union all parcel geometry and buffer by 1 to get adjacent parcels

```
Geometry[] geom = new Geometry[geometries.size()];  
geometries.toArray(geom);  
GeometryFactory fact = geom[0].getFactory();  
Geometry geomColl = fact.createGeometryCollection(geom);  
Geometry union = geomColl.buffer(0);  
Geometry bufferedGeometry = union.buffer(1);
```

- ❑ Perform another spatial query using the buffered geometry to get parcels

# CountyView Web: Reverse Geocode

The screenshot displays the CountyView Web application interface within a Mozilla Firefox browser window. The browser's address bar shows the URL: `http://shasta5.co.pierce.wa.us/imf/jsp?site=prc`. The application header includes the Pierce County logo and the text "Pierce County CountyView Web".

The main content area is titled "Reverse Geocode" and displays the following information:

- Left Address: 16211 CANYON RD E
- Right Address: 16212 CANYON RD E

The central part of the interface is a map of Pierce County, Washington, showing a grid of streets. A blue star icon marks the location of 16211 Canyon Rd E and 16212 Canyon Rd E. The map includes labels for various streets such as 12th St E, 14th St E, 16th St E, 18th St E, 20th St E, 22nd St E, 24th St E, 26th St E, 28th St E, 30th St E, 32nd St E, 34th St E, 36th St E, 38th St E, 40th St E, 42nd St E, 44th St E, 46th St E, 48th St E, 50th St E, 52nd St E, 54th St E, 56th St E, 58th St E, 60th St E, 62nd St E, 64th St E, 66th St E, 68th St E, 70th St E, 72nd St E, 74th St E, 76th St E, 78th St E, 80th St E, 82nd St E, 84th St E, 86th St E, 88th St E, 90th St E, 92nd St E, 94th St E, 96th St E, 98th St E, 100th St E, 102nd St E, 104th St E, 106th St E, 108th St E, 110th St E, 112th St E, 114th St E, 116th St E, 118th St E, 120th St E, 122nd St E, 124th St E, 126th St E, 128th St E, 130th St E, 132nd St E, 134th St E, 136th St E, 138th St E, 140th St E, 142nd St E, 144th St E, 146th St E, 148th St E, 150th St E, 152nd St E, 154th St E, 156th St E, 158th St E, 160th St E, 162nd St E, 164th St E, 166th St E, 168th St E, 170th St E, 172nd St E, 174th St E, 176th St E, 178th St E, 180th St E, 182nd St E, 184th St E, 186th St E, 188th St E, 190th St E, 192nd St E, 194th St E, 196th St E, 198th St E, 200th St E, 202nd St E, 204th St E, 206th St E, 208th St E, 210th St E, 212th St E, 214th St E, 216th St E, 218th St E, 220th St E, 222nd St E, 224th St E, 226th St E, 228th St E, 230th St E, 232nd St E, 234th St E, 236th St E, 238th St E, 240th St E, 242nd St E, 244th St E, 246th St E, 248th St E, 250th St E, 252nd St E, 254th St E, 256th St E, 258th St E, 260th St E, 262nd St E, 264th St E, 266th St E, 268th St E, 270th St E, 272nd St E, 274th St E, 276th St E, 278th St E, 280th St E, 282nd St E, 284th St E, 286th St E, 288th St E, 290th St E, 292nd St E, 294th St E, 296th St E, 298th St E, 300th St E, 302nd St E, 304th St E, 306th St E, 308th St E, 310th St E, 312nd St E, 314th St E, 316th St E, 318th St E, 320th St E, 322nd St E, 324th St E, 326th St E, 328th St E, 330th St E, 332nd St E, 334th St E, 336th St E, 338th St E, 340th St E, 342nd St E, 344th St E, 346th St E, 348th St E, 350th St E, 352nd St E, 354th St E, 356th St E, 358th St E, 360th St E, 362nd St E, 364th St E, 366th St E, 368th St E, 370th St E, 372nd St E, 374th St E, 376th St E, 378th St E, 380th St E, 382nd St E, 384th St E, 386th St E, 388th St E, 390th St E, 392nd St E, 394th St E, 396th St E, 398th St E, 400th St E, 402nd St E, 404th St E, 406th St E, 408th St E, 410th St E, 412nd St E, 414th St E, 416th St E, 418th St E, 420th St E, 422nd St E, 424th St E, 426th St E, 428th St E, 430th St E, 432nd St E, 434th St E, 436th St E, 438th St E, 440th St E, 442nd St E, 444th St E, 446th St E, 448th St E, 450th St E, 452nd St E, 454th St E, 456th St E, 458th St E, 460th St E, 462nd St E, 464th St E, 466th St E, 468th St E, 470th St E, 472nd St E, 474th St E, 476th St E, 478th St E, 480th St E, 482nd St E, 484th St E, 486th St E, 488th St E, 490th St E, 492nd St E, 494th St E, 496th St E, 498th St E, 500th St E, 502nd St E, 504th St E, 506th St E, 508th St E, 510th St E, 512nd St E, 514th St E, 516th St E, 518th St E, 520th St E, 522nd St E, 524th St E, 526th St E, 528th St E, 530th St E, 532nd St E, 534th St E, 536th St E, 538th St E, 540th St E, 542nd St E, 544th St E, 546th St E, 548th St E, 550th St E, 552nd St E, 554th St E, 556th St E, 558th St E, 560th St E, 562nd St E, 564th St E, 566th St E, 568th St E, 570th St E, 572nd St E, 574th St E, 576th St E, 578th St E, 580th St E, 582nd St E, 584th St E, 586th St E, 588th St E, 590th St E, 592nd St E, 594th St E, 596th St E, 598th St E, 600th St E, 602nd St E, 604th St E, 606th St E, 608th St E, 610th St E, 612nd St E, 614th St E, 616th St E, 618th St E, 620th St E, 622nd St E, 624th St E, 626th St E, 628th St E, 630th St E, 632nd St E, 634th St E, 636th St E, 638th St E, 640th St E, 642nd St E, 644th St E, 646th St E, 648th St E, 650th St E, 652nd St E, 654th St E, 656th St E, 658th St E, 660th St E, 662nd St E, 664th St E, 666th St E, 668th St E, 670th St E, 672nd St E, 674th St E, 676th St E, 678th St E, 680th St E, 682nd St E, 684th St E, 686th St E, 688th St E, 690th St E, 692nd St E, 694th St E, 696th St E, 698th St E, 700th St E, 702nd St E, 704th St E, 706th St E, 708th St E, 710th St E, 712nd St E, 714th St E, 716th St E, 718th St E, 720th St E, 722nd St E, 724th St E, 726th St E, 728th St E, 730th St E, 732nd St E, 734th St E, 736th St E, 738th St E, 740th St E, 742nd St E, 744th St E, 746th St E, 748th St E, 750th St E, 752nd St E, 754th St E, 756th St E, 758th St E, 760th St E, 762nd St E, 764th St E, 766th St E, 768th St E, 770th St E, 772nd St E, 774th St E, 776th St E, 778th St E, 780th St E, 782nd St E, 784th St E, 786th St E, 788th St E, 790th St E, 792nd St E, 794th St E, 796th St E, 798th St E, 800th St E, 802nd St E, 804th St E, 806th St E, 808th St E, 810th St E, 812nd St E, 814th St E, 816th St E, 818th St E, 820th St E, 822nd St E, 824th St E, 826th St E, 828th St E, 830th St E, 832nd St E, 834th St E, 836th St E, 838th St E, 840th St E, 842nd St E, 844th St E, 846th St E, 848th St E, 850th St E, 852nd St E, 854th St E, 856th St E, 858th St E, 860th St E, 862nd St E, 864th St E, 866th St E, 868th St E, 870th St E, 872nd St E, 874th St E, 876th St E, 878th St E, 880th St E, 882nd St E, 884th St E, 886th St E, 888th St E, 890th St E, 892nd St E, 894th St E, 896th St E, 898th St E, 900th St E, 902nd St E, 904th St E, 906th St E, 908th St E, 910th St E, 912nd St E, 914th St E, 916th St E, 918th St E, 920th St E, 922nd St E, 924th St E, 926th St E, 928th St E, 930th St E, 932nd St E, 934th St E, 936th St E, 938th St E, 940th St E, 942nd St E, 944th St E, 946th St E, 948th St E, 950th St E, 952nd St E, 954th St E, 956th St E, 958th St E, 960th St E, 962nd St E, 964th St E, 966th St E, 968th St E, 970th St E, 972nd St E, 974th St E, 976th St E, 978th St E, 980th St E, 982nd St E, 984th St E, 986th St E, 988th St E, 990th St E, 992nd St E, 994th St E, 996th St E, 998th St E, 1000th St E.

The map also shows major roads like BINGHAM AVE, CANYON RD E, MILITARY RD E, and SUNRISE BLVD E. A purple line is visible on the right side of the map, and a blue star marks the location of 16211 Canyon Rd E and 16212 Canyon Rd E.

The bottom of the interface features a status bar with the following information:

- Scale: 1:32,441
- Extents: Select a location
- Map Tool: Reverse Geocode
- Active Layer: \* NO ACTIVE LAYER \*
- Cursor Location: NAD 1983 HARN STATEPLANE: 1188873.9769737746, 634139.831060275

The application is powered by Arcocortex.

# CountyView Web: Reverse Geocode

---

- ❑ Perform spatial query around an  $x,y$  at some distance on roads layer.
- ❑ Get the closest Geometry (LineString)

```
Coordinate coordinate = new Coordinate(x,y);
```

```
// Put the user's Coordinate on the LineString
```

```
LocationIndexedLine lineRef = new LocationIndexedLine(lineString);
```

```
LinearLocation loc = lineRef.project(coordinate);
```

```
Coordinate coordOnLine = loc.getCoordinate(lineString);
```

```
// Figure out how far the Coordinate is along the LineString
```

```
LengthLocationMap locationMap = new LengthLocationMap(lineString);
```

```
double distanceAlong = locationMap.getLength(loc);
```

```
double lineLength = lineString.getLength();
```

```
double percentAlong = distanceAlong / lineLength;
```

```
// Use percentAlong to interpolate between to and from address ranges
```

# CountyView Web: Profiles

The screenshot displays the CountyView Web interface in Mozilla Firefox. The main window shows a map with a red profile line drawn across it. A sidebar on the left contains the 'Create a Profile Tool' instructions and buttons. A secondary window titled 'CountyView Web :: Profile' is open on the right, showing a graph of Elevation vs. Distance and a data table.

**Create a Profile Tool**

4 points recorded. Click the map to add more points to the line, or click the OK button if you are finished drawing your line.

To restart the line, click the Clear button. Your line is 4745.705425376791 ft in length.

**Profile**

Add Contours | Save As PDF

**Profile**

Elevation

Distance

X	Y	Elevation	Distance
1	1189112.9680998023	640245.2077257575	436.0
2	1189162.3197811628	640245.7782654264	438.0
3	1189193.3789828771	640246.1373313422	440.0
4	1189228.8178750013	640246.547029517	442.0
5	1189253.7193403884	640246.8349077295	444.0
6	1189339.5644318066	640247.8273365321	446.0
7	1189434.681014117	640248.9269502005	448.0
8	1189460.844507165	640249.2294183281	450.0
9	1189491.2814442161	640249.5812904327	452.0
10	1189518.4462703145	640249.8953346651	454.0
11	1189535.059003973	640250.0873893895	456.0
12	1189555.9860548105	640250.3293206131	458.0
13	1189573.228657044	640250.5286570551	460.0
14	1189587.961484561	640250.6989787605	462.0
15	1189600.128841343	640250.8396418446	464.0
16	1189609.863523499	640250.9521815442	466.0
17	1189620.8033077405	640251.0786530168	468.0
18	1189627.7357036886	640251.1587963226	470.0

Scale: 1:5,710 Extents: Select a location Map Tool: Create a profile. Active Layer: \* NO ACTIVE LAYER \*

# CountyView Web: Profiles (code)

---

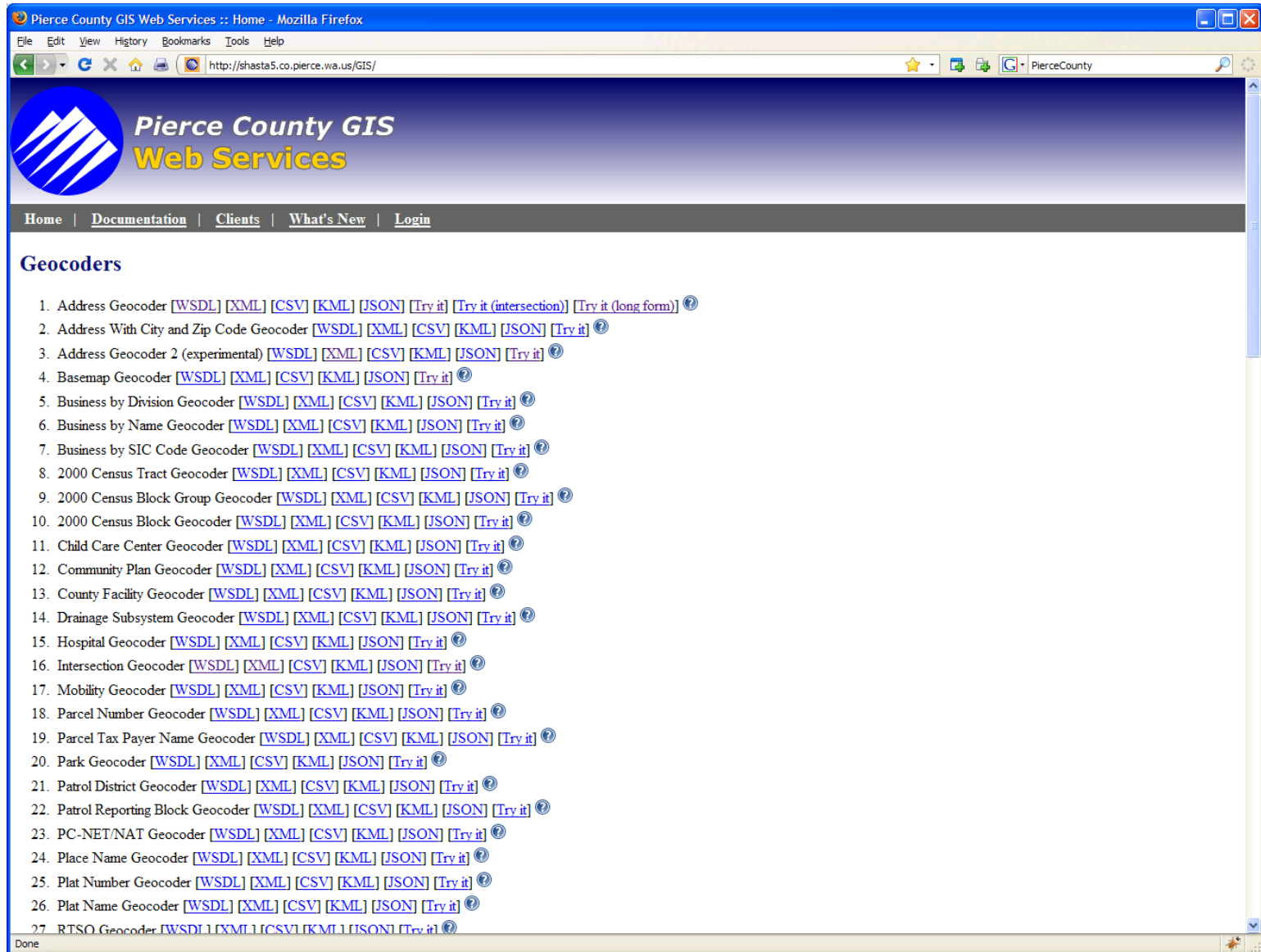
- ❑ Perform spatial query getting all contours that intersect with a LineString draw by the user.

```
// Index the profile LineString for linear referencing
LocationIndexedLine lineRef = new LocationIndexedLine(lineString);
double lineLength = lineString.getLength();

// For each contour, find the intersection between the users LineString and
// the contour
Geometry intersection = lineString.intersection(geometry);

// Get the coordinate of this intersection and calculate the distance along and percent
// long the LineString
Coordinate coordinate = ((Point)intersection.getGeometryN(i)).getCoordinate();
LinearLocation loc = lineRef.project(coordinate);
LengthLocationMap locationMap = new LengthLocationMap(lineString);
double distanceAlong = locationMap.getLength(loc);
double percentAlong = distanceAlong / lineLength;
```

# GIS Web Services



The screenshot shows a Mozilla Firefox browser window displaying the Pierce County GIS Web Services website. The browser's address bar shows the URL <http://shasta5.co.pierce.wa.us/GIS/>. The website header features the Pierce County GIS logo, which consists of a blue circle with white diagonal lines, and the text "Pierce County GIS Web Services". Below the header is a navigation menu with links for Home, Documentation, Clients, What's New, and Login. The main content area is titled "Geocoders" and lists 27 different geocoding services, each with links to its WSDL, XML, CSV, KML, and JSON endpoints, and a "Try it" button. The services listed are:

1. Address Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it] [Try it (intersection)] [Try it (long form)]
2. Address With City and Zip Code Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
3. Address Geocoder 2 (experimental) [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
4. Basemap Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
5. Business by Division Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
6. Business by Name Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
7. Business by SIC Code Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
8. 2000 Census Tract Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
9. 2000 Census Block Group Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
10. 2000 Census Block Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
11. Child Care Center Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
12. Community Plan Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
13. County Facility Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
14. Drainage Subsystem Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
15. Hospital Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
16. Intersection Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
17. Mobility Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
18. Parcel Number Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
19. Parcel Tax Payer Name Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
20. Park Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
21. Patrol District Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
22. Patrol Reporting Block Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
23. PC-NET/NAT Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
24. Place Name Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
25. Plat Number Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
26. Plat Name Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]
27. RTSO Geocoder [WSDL] [XML] [CSV] [KML] [JSON] [Try it]

The browser window also shows standard navigation buttons (back, forward, home, stop, refresh) and a search bar in the top right corner. The status bar at the bottom of the browser window displays "Done".

# GIS Web Services

---

- Pierce County GIS web services
  - Geocoders, Spatial Queries, Projection, Open @, Tile Caches, Data
  - SOAP, XML, CSV, JSON, KML
  
- Documentation and Examples
  - Suggested user interfaces for non GIS developers
  
- Serves thousands of requests to 12 applications across 3 departments.

# GIS Web Services

---

- Common Geometry model between ArcSDE and ArcIMS for Spatial Queries
- Reprojection
  - JTS and GeoTools

# GIS Web Services: Spatial Queries

---

## □ Buffer service

- Parameters Point, Distance, Layer
- Buffer the Point by the Distance
- Turn buffered Polygon into ArcIMS AXL or ArcSDE Geometry (or WKT for PostGIS)
- Perform spatial query
- Turn AXL, ArcSDE Geometry, WKT back into JTS Geometry

# GIS Web Services: Projection

---

- Uses JTS and GeoTools
- Uses European Petroleum Survey Group (EPSG)
  - EPSG:4326 is WGS 84
  - EPSG:2927 is WA State Plane South (feet)

```
CoordinateReferenceSystem fromCRS = CRS.decode(fromEPSG);  
CoordinateReferenceSystem toCRS = CRS.decode(toEPSG);
```

```
MathTransform math = CRS.findMathTransform(fromCRS,toCRS);  
DirectPosition pos = new GeneralDirectPosition(x, y);  
DirectPosition geoPos = math.transform(pos,null);
```

# JTSIO

---

- Java Library for reading and writing to and from JTS Geometry Objects and geometry string formats
  
- JTS to AXL
  - Point(10,20) to <POINT x="10.0" y="20.0" />
  
- Other Neogeography/Web 2.0 geometry string formats
  - KML
  - GeoJSON
  - GeoRSS
  - GML
  - GPX
  - WKT

# Format Examples

---

## **WKT**

```
POINT(1 1)
```

## **AXL**

```
<POINT x="1.0" y="1.0" />
```

## **GeoJSON**

```
{"type":"Point","coordinates":[1,1]}
```

## **GeoRSS**

```
<georss:point>1.0 1.0</georss:point>
```

## **GML**

```
<gml:Point xmlns:gml="http://www.opengis.net/gml">  
  <gml:coordinates>1.0,1.0</gml:coordinates>  
</gml:Point>
```

## **GPX**

```
<wpt xmlns="http://www.topografix.com/GPX/1/1" lat="1.0" lon="1.0" />
```

## **KML**

```
<Point><coordinates>1.0,1.0</coordinates></Point>
```

# JTSIO Geometry/Format Matrix

	axl	geo Json	geo JsonDoc	Geo Rss	Geo Rss Doc	Geo Rss Gml	Geo Rss Gml Doc	gml	gmlDoc	gpx	Gpx Doc	kml	Kml Doc	wkt
Point	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Multi Point	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
Line String	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Linear Ring	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Polygon	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE
Multi LineString	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Multi Polygon	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE
Geometry Collection	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE

# JTSIO in action

---

```
GeometryFactory geometryFactory = new GeometryFactory();  
Coordinate coord = new Coordinate(5.1, 5.2);  
Point point = geometryFactory.createPoint(coord);
```

```
GeoJsonJtsWriter writer = new GeoJsonJtsWriter();  
String geoJson = writer.writePointToString(point);
```

```
{"type":"Point","coordinates":[5.1,5.2]}
```

```
GeoJsonJtsReader reader = new GeoJsonJtsReader();  
Point point2 = reader.readPoint(geoJson);
```

# JTS Web Processing

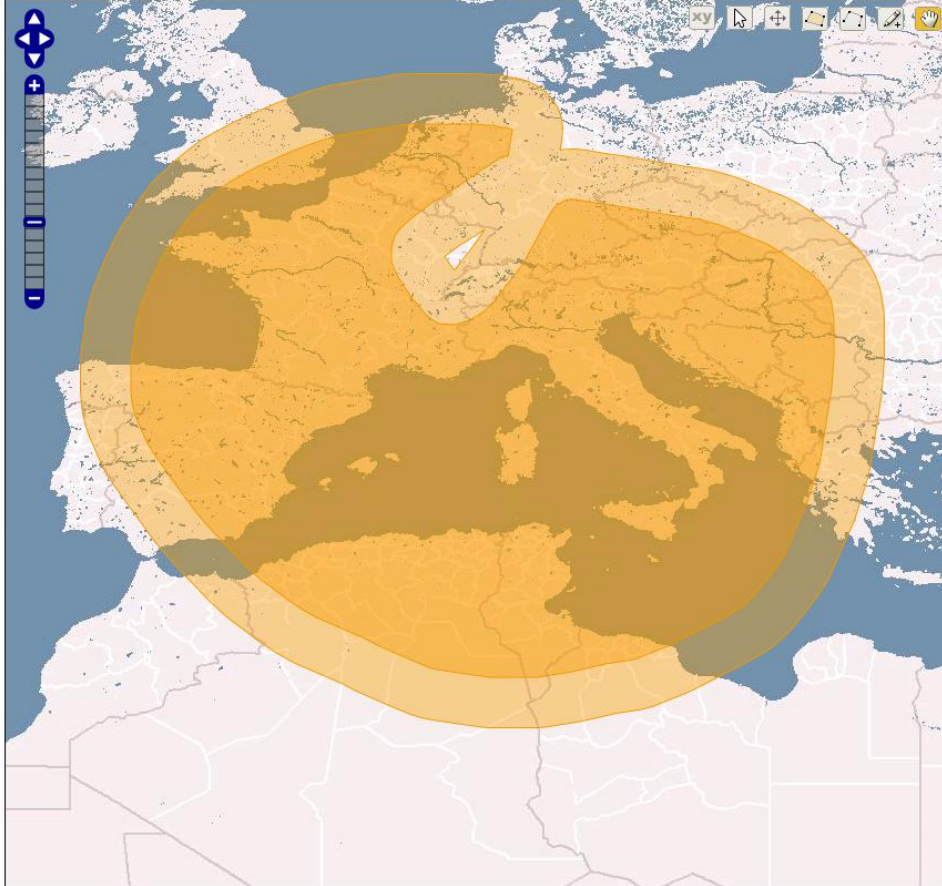
JTS Web Processing :: JTS Web Processing :: OpenLayers Demo - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://whitneydev.co.pierce.wa.us/jts/map.htm PierceCounty

## JTS Web Processing

[Home](#) [Services](#) [Map](#) [Convert](#) [Convert with reprojection](#) [Geometry in all Formats](#) [Geometries in a Format](#) [Geometry/Format Matrix](#)



JTS Web Processing

- [Clear Features](#)
- [Buffer](#)  
Distance:   
Quadrant Segments:   
Cap Style:
- [Contains](#)
- [Convex Hull](#)
- [Coordinate at Angle](#)  
Angle:   
Distance:
- [Coordinates at Angles](#)  
Angle:   
Distance(s):
- [Covered By](#)
- [Covers](#)
- [Create LineString Arc](#)  
Distance:   
Start Angle:   
End Angle:   
Tolerance:
- [Create Quad Strip](#)
- [Create Triangle Fan](#)
- [Create Triangle Strip](#)
- [Crosses](#)
- [Difference](#)
- [Disjoint](#)
- [Distance](#)
- [Distance LineString](#)
- [Douglas-Peucker Simplifier](#)  
Distance Tolerance:
- [Draw WKT](#)

Done

# JTS Web Processing

---

- RESTful Geoprocessing Server that exposes JTS spatial operators
  - Buffer, intersection, union, touches, ect...
- Geometry had to be encoded in text based format using JTSIO
- Idea based on Christopher Schmidt's WebProcessingServer
- 40 or so web services
- Consumable from Javascript via Ajax
- Displayable in OpenLayers

# JTS Web Processing: How it works

---

## Request

```
http://whitneydev.co.pierce.wa.us/jts/services/wkt/buffer?  
geometry=POINT (108420.33  
753808.59)&distance=500&quadrantSegments=8&capStyle=r  
ound&sourceCRS=EPSG:2927&targetCRS=EPSG:2927
```

## Response

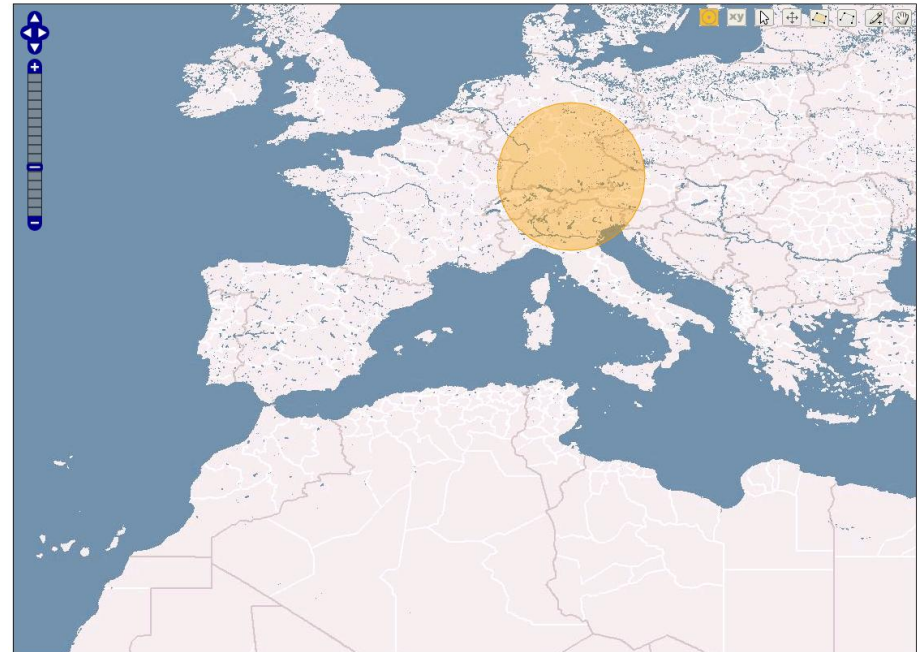
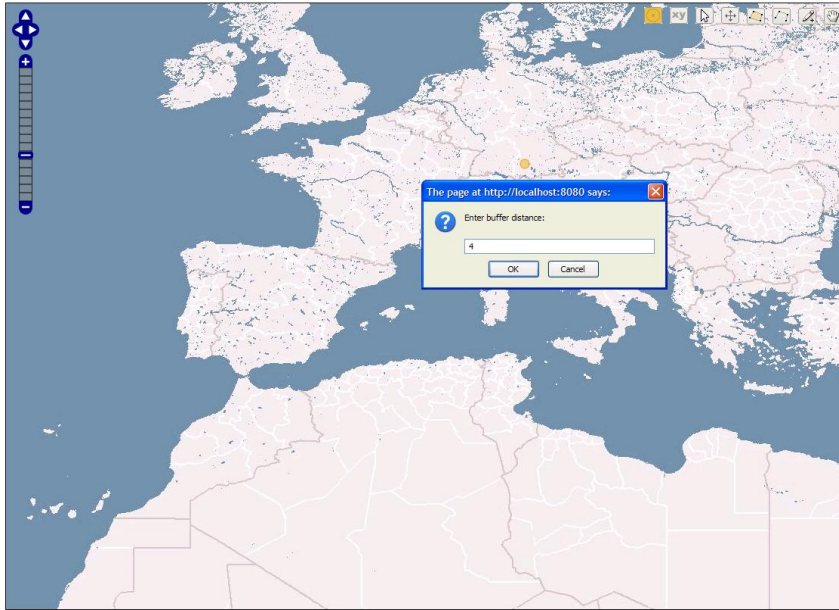
```
POLYGON ((108920.33 753808.59, 108910.72264020162 753711.044838992,  
108882.26976625565 753617.2482838174, 108836.06480615128 753530.8048834902,  
108773.88339059327 753455.0366094067, 108698.1151165098 753392.8551938487,  
108611.67171618255 753346.6502337443, 108517.87516100807 753318.1973597984,  
108420.33 753308.59, 108322.78483899194 753318.1973597984, 108228.98828381745  
753346.6502337443, 108142.5448834902 753392.8551938487, 108066.77660940673  
753455.0366094067, 108004.59519384873 753530.8048834902, 107958.39023374436  
753617.2482838174, 107929.93735979838 753711.044838992, 107920.33 753808.59,  
107929.93735979838 753906.135161008, 107958.39023374436 753999.9317161825,  
108004.59519384873 754086.3751165097, 108066.77660940673 754162.1433905932,  
108142.5448834902 754224.3248061512, 108228.98828381745 754270.5297662556,  
108322.78483899194 754298.9826402016, 108420.33 754308.59, 108517.87516100807  
754298.9826402016, 108611.67171618255 754270.5297662556, 108698.1151165098  
754224.3248061512, 108773.88339059327 754162.1433905932, 108836.06480615128  
754086.3751165097, 108882.26976625565 753999.9317161825, 108910.72264020162  
753906.135161008, 108920.33 753808.59))
```

# JTS Web Processing and OpenLayers

---

```
/**
 * Create a custom buffer point tool
 */
function createBufferPointTool() {
  var tool = new OpenLayers.Control();
  OpenLayers.Util.extend(tool, {
    draw: function () {
      this.handler = new OpenLayers.Handler.Point(tool, {"done": this.notice});
    },
    notice: function (pt) {
      var distanceStr = prompt("Enter buffer distance:");
      if (!distanceStr || isNaN(distanceStr)) {
        alert("Please enter a distance!");
        return;
      }
      var distance = parseFloat(distanceStr);
      var geoJsonFormat = new OpenLayers.Format.GeoJSON();
      var geoJson = geoJsonFormat.write(pt, false);
      $.ajax({
        type: "POST",
        url: "/jts/services/geoJson/buffer",
        data: {
          "geometry": geoJson,
          "distance": distance
        },
        dataType: 'json',
        success: function(json){
          var geoJsonFormat = new OpenLayers.Format.GeoJSON();
          var feature = geoJsonFormat.read(json);
          if (json) {
            vlayer.addFeatures(feature);
          }
        },
        error: function(request, text, error) {
          alert("Error buffering point!");
        }
      });
    },
    CLASS_NAME: "OpenLayers.Control.BufferPointTool",
    displayClass: "olControlBufferPointTool",
    type: OpenLayers.Control.TYPE_TOOL
  });
  return tool;
}
```

# JTS Web Processing and OpenLayers



# Conclusion

---

- ❑ GIS Software Development doesn't have to be either (ESRI) or (Open Source)
- ❑ JTS is an incredibly powerful and useful library
- ❑ JTS can easily extend ArcIMS and ArcSDE