Application: gvSIG desktop - gvSIG bugs #546
Error in class Converter with one solution

04/18/2012 08:26 PM - Leticia Riestra

Status:	Closed	**% Done:**	100%
Priority:	Normal	**Spent time:**	0.00 hour
Assignee:	Joaquín del Cerro Murciano		
Category:			
Target version:	2.0.0-devel-2048		
Severity:		**Add-on version:**	
gvSIG version:	2.0.0	**Add-on build:**	
gvSIG build:	2045	**Add-on resolve version:**	
Operative System:		**Add-on resolve build:**	
Keywords:		**Proyecto:**	
Has patch:	Yes	**Hito:**	
Add-on name:	Unknown		

Description

<English>
We have detected some bugs in class Converter when we work with multipoint, multisurface or multicurve
We have a proposal with some corrections. This code is tested and is being used in an application now in operation (gisEIEL 2.3) and other application about to release (gisEIEL 3.0 based on gvSIG 2.0)

- The method jtsToGeometry failure if the type geometry is MultiPoint

```
if (geo.isEmpty()) {
    shpNew = null;
}

try{
    *if (geo instanceof com.vividsolutions.jts.geom.MultiPoint) {
        shpNew = geomManager.create(TYPES.MULTIPOINT, SUBTYPES.GEOM2D);
        for (int i = 0; i < geo.getNumGeometries(); i++) {
            Point point = (Point) geo.getGeometryN(i);
            ((MultiPoint)shpNew).addPoint((org.gvsig.fmap.geom.primitive.Point) Converter.jtsToGeometry(point));
        }

    }*

    if (geo instanceof Polygon) {
        shpNew = geomManager.createSurface(toShape((Polygon) geo), SUBTYPES.GEOM2D);
    }

    ....
```

- Changes in the method *com.vividsolutions.jts.geom.Geometry geometryToJts (Geometry shp, shapeType int, int srid)* to redefine the case of multipoint, and MultiCurve MultiSurface and to fix a bug in the case of ellipse

ELLIPSE

```
case Geometry.TYPES.ELLIPSE:
```

```
arrayLines = new ArrayList();

ArrayList shells = new ArrayList();
ArrayList holes = new ArrayList();
Coordinate[] points = null;

theIterator = shp.getPathIterator(null, manager.getFlatness());

while (!theIterator.isDone()) {
   //while not done
   theType = theIterator.currentSegment(theData);

   //Populate a segment of the new
   // GeneralPathX object.
   //Process the current segment to populate a new
   // segment of the new GeneralPathX object.
   switch (theType) {
   case PathIterator.SEG_MOVETO:

      // System.out.println("SEG_MOVETO");
      if (arrayCoords == null) {
         arrayCoords = new ArrayList();
      } else {
         points = CoordinateArrays.toCoordinateArray(arrayCoords);

         try {
            LinearRing ring = geomFactory.createLinearRing(points);

            if (CGAlgorithms.isCCW(points)) {
               holes.add(ring);
            } else {
               shells.add(ring);
            }
         } catch (Exception e) {
            /* (jaume) caso cuando todos los puntos son iguales
             * devuelvo el propio punto
             */
            boolean same = true;
            for (int i = 0; i < points.length-1 && same; i++) {
               if (points[i].x != points[i+1].x ||
                     points[i].y != points[i+1].y /*||
                     points[i].z != points[i+1].z*/
               ) {
                  same = false;
               }
            }
            if (same) {
               return geomFactory.createPoint(points[0]);
            }
            /*
             * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
             * una linea
             */
```

```
            if (points.length>1 && points.length<=3) {
               // return geomFactory.createLineString(points);
               return geomFactory.createMultiLineString(new LineString[] {geomFactory.createLineString(points)});
            }

            System.err.println(
            "Caught Topology exception in GMLLinearRingHandler");

            return null;
         }

         /* if (numParts == 1)
          {
          linRingExt = new GeometryFactory().createLinearRing(
          CoordinateArrays.toCoordinateArray(arrayCoords));
          }
          else
          {
          linRing = new GeometryFactory().createLinearRing(
          CoordinateArrays.toCoordinateArray(arrayCoords));
          arrayLines.add(linRing);
          } */
         arrayCoords = new ArrayList();
      }

      numParts++;
      arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

      break;

   case PathIterator.SEG_LINETO:

      // System.out.println("SEG_LINETO");
      arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

      break;

   case PathIterator.SEG_QUADTO:
      System.out.println("SEG_QUADTO Not supported here");

      break;

   case PathIterator.SEG_CUBICTO:
      System.out.println("SEG_CUBICTO Not supported here");

      break;

   case PathIterator.SEG_CLOSE:

      // Aï¿½adimos el primer punto para cerrar.
      Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
```

```
            arrayCoords.add(new Coordinate(firstCoord.x,
                    firstCoord.y));

            break;
        } //end switch

        // System.out.println("theData[0] = " + theData[0] + " theData[1]=" + theData[1]);
        theIterator.next();
    } //end while loop

    Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
    Coordinate lastCoord = (Coordinate) arrayCoords.get(arrayCoords
            .size() - 1);
    if (!isClosed(firstCoord, lastCoord)) {
        arrayCoords.add(firstCoord);
    }
    points = CoordinateArrays.toCoordinateArray(arrayCoords);

    try {
        LinearRing ring = geomFactory.createLinearRing(points);

        if (CGAlgorithms.isCCW(points)) {
            holes.add(ring);
        } else {
            shells.add(ring);
        }
        ring.setSRID(srid);
    } catch (Exception e) {
        /* (jaume) caso cuando todos los puntos son iguales
         * devuelvo el propio punto
         */
        boolean same = true;
        for (int i = 0; i < points.length-1 && same; i++) {
            if (points[i].x != points[i+1].x ||
                    points[i].y != points[i+1].y /*||
                    points[i].z != points[i+1].z*/
            ) {
                same = false;
            }
        }
        if (same) {
            geoJTS = geomFactory.createPoint(points[0]);
            geoJTS.setSRID(srid);
            return geoJTS;
        }
        /*
         * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
         * una linea
         */
        if (points.length>1 && points.length<=3) {
            // return geomFactory.createLineString(points);
            geoJTS = geomFactory
            .createMultiLineString(new LineString[] { geomFactory
```

```
                .createLineString(points) });
        geoJTS.setSRID(srid);
        return geoJTS;
    }
    System.err.println(
    "Caught Topology exception in GMLLinearRingHandler");

    return null;
}

/* linRing = new GeometryFactory().createLinearRing(
 CoordinateArrays.toCoordinateArray(arrayCoords)); */

// System.out.println("NumParts = " + numParts);
//now we have a list of all shells and all holes
ArrayList holesForShells = new ArrayList(shells.size());

for (int i = 0; i < shells.size(); i++) {
    holesForShells.add(new ArrayList());
}

//find homes
for (int i = 0; i < holes.size(); i++) {
    LinearRing testRing = (LinearRing) holes.get(i);
    LinearRing minShell = null;
    Envelope minEnv = null;
    Envelope testEnv = testRing.getEnvelopeInternal();
    Coordinate testPt = testRing.getCoordinateN(0);
    LinearRing tryRing = null;

    for (int j = 0; j < shells.size(); j++) {
        tryRing = (LinearRing) shells.get(j);

        Envelope tryEnv = tryRing.getEnvelopeInternal();

        if (minShell != null) {
            minEnv = minShell.getEnvelopeInternal();
        }

        boolean isContained = false;
        Coordinate[] coordList = tryRing.getCoordinates();

        if (tryEnv.contains(testEnv) &&
                (CGAlgorithms.isPointInRing(testPt, coordList) ||
                        (pointInList(testPt, coordList)))) {
            isContained = true;
        }

        // check if this new containing ring is smaller than the current minimum ring
        if (isContained) {
            if ((minShell == null) || minEnv.contains(tryEnv)) {
                minShell = tryRing;
            }
```

```
            }
        }

        if (minShell == null) {
            //                  System.out.println(
            //                  "polygon found with a hole thats not inside a shell");
            //                  azabala: we do the assumption that this hole is really a shell (polygon)
            //                  whose point werent digitized in the right order
            Coordinate[] cs = testRing.getCoordinates();
            Coordinate[] reversed = new Coordinate[cs.length];
            int pointIndex = 0;
            for(int z = cs.length-1; z >= 0; z--){
                reversed[pointIndex] = cs[z];
                pointIndex++;
            }
            LinearRing newRing = geomFactory.createLinearRing(reversed);
            shells.add(newRing);
            holesForShells.add(new ArrayList());
        } else {
            ((ArrayList) holesForShells.get(shells.indexOf(minShell))).add(testRing);
        }
    }

    Polygon[] polygons = new Polygon[shells.size()];

    for (int i = 0; i < shells.size(); i++) {
        polygons[i] = geomFactory.createPolygon((LinearRing) shells.get(
                i),
                (LinearRing[]) ((ArrayList) holesForShells.get(i)).toArray(
                        new LinearRing[0]));
        polygons[i].setSRID(srid);
    }
    // CAMBIO: ENTREGAMOS SIEMPRE MULTILINESTRING, QUE ES
    // LO QUE HACE TODO EL MUNDO CUANDO ESCRIBE EN POSTGIS
    // O CON GEOTOOLS
    // if (numParts > 1) // Generamos una MultiLineString

    /* if (polygons.length == 1) {
     return polygons[0];
     } */

    // FIN CAMBIO

    holesForShells = null;
    shells = null;
    holes = null;

    /*if (polygons.length == 1) {
        geoJTS = polygons[0];
    } else {
        // its a multi part
        geoJTS = geomFactory.createMultiPolygon(polygons);
    }
```

```
            geoJTS.setSRID(srid);*/
            //its a multi part
            geoJTS = geomFactory.createMultiPolygon(polygons);
            geoJTS.setSRID(srid);

            /* if (numParts > 1) // Generamos un Polygon con agujeros
             {
             arrayLines.add(linRing);
             // geoJTS = new GeometryFactory().createPolygon(linRingExt,
              // GeometryFactory.toLinearRingArray(arrayLines));
               geoJTS = new GeometryFactory().buildGeometry(arrayLines);

              // geoJTS = Polygonizer.class.
               }
               else
               {
               geoJTS = new GeometryFactory().createPolygon(linRing,null);
               } */
            break;
```

MULTICURVE, MULTIPOINT, MULTISURFACE

```
    case Geometry.TYPES.MULTICURVE:
            geoJTS = multiCurveToJts((MultiCurve)shp, srid);
            geoJTS.setSRID(srid);
            break;

        case Geometry.TYPES.MULTIPOINT:
            geoJTS = geometryToJts((MultiPoint)shp);
            geoJTS.setSRID(srid);
            break;

        case Geometry.TYPES.MULTISURFACE:
            geoJTS = multiSurfaceToJts((MultiSurface)shp, srid);
            geoJTS.setSRID(srid);
            break;
```

- Creation of new methods to support the above changes

```
    public static com.vividsolutions.jts.geom.Geometry multiCurveToJts(MultiCurve geom, int srid) {
        LineString[] lines = new LineString[geom.getPrimitivesNumber()];
        for (int i = 0; i < lines.length; i++){
            lines[i] = (LineString) curveToJts((geom.getPrimitiveAt(i)), srid);
        }
        return new com.vividsolutions.jts.geom.GeometryFactory().createMultiLineString(lines);
      }

      private static com.vividsolutions.jts.geom.Geometry curveToJts(Geometry shp, int srid){
```

```
ArrayList arrayLines;
LineString lin = null;
PathIterator theIterator;
int theType;
int numParts = 0;
double[] theData = new double[6];
ArrayList arrayCoords = null;
Coordinate coord;

arrayLines = new ArrayList();
theIterator = shp.getPathIterator(null, manager.getFlatness());

while (!theIterator.isDone()) {
   //while not done
   theType = theIterator.currentSegment(theData);

   //Populate a segment of the new
   // GeneralPathX object.
   //Process the current segment to populate a new
   // segment of the new GeneralPathX object.
   switch (theType) {
   case PathIterator.SEG_MOVETO:

      // System.out.println("SEG_MOVETO");
      if (arrayCoords == null) {
         arrayCoords = new ArrayList();
      } else {
         lin = geomFactory.createLineString(CoordinateArrays.toCoordinateArray(
               arrayCoords));
         lin.setSRID(srid);
         arrayLines.add(lin);
         arrayCoords = new ArrayList();
      }

      numParts++;
      coord = new Coordinate(theData[0], theData[1]);

      arrayCoords.add(coord);

      break;

   case PathIterator.SEG_LINETO:

      // System.out.println("SEG_LINETO");
      arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

      break;

   case PathIterator.SEG_QUADTO:
      System.out.println("Not supported here");

      break;
```

```
            case PathIterator.SEG_CUBICTO:
                System.out.println("Not supported here");

                break;

            case PathIterator.SEG_CLOSE:
                // Aï¿½adimos el primer punto para cerrar.
                Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
                // Solo anyadimos cuando no esta ya cerrado
                arrayCoords.add(new Coordinate(firstCoord.x,
                        firstCoord.y));

                break;
            } //end switch

            theIterator.next();
        } //end while loop

        if (arrayCoords.size()<2) {

        }else{

            lin = new com.vividsolutions.jts.geom.GeometryFactory().createLineString(CoordinateArrays.toCoordinateArray(
                    arrayCoords));

            lin.setSRID(srid);

        }

        return lin;

    }



/**
     * Método creado para construir un MultiSurface formado por varias surface
     *
     * @author Leticia Riestra
     * @param geom
     * @return
     */
    public static com.vividsolutions.jts.geom.Geometry multiSurfaceToJts(MultiSurface geom, int srid) {
        Polygon[] polygons = new Polygon[geom.getPrimitivesNumber()];
        for (int i = 0; i < polygons.length; i++){
            MultiPolygon polygon = (MultiPolygon)surfaceToJts((geom.getPrimitiveAt(i)), srid);

            polygons[i] = (Polygon)polygon.getGeometryN(0);//(Polygon) surfaceToJts((geom.getPrimitiveAt(i)), srid);
        }
        return new com.vividsolutions.jts.geom.GeometryFactory().createMultiPolygon(polygons);
    }
```

```
private static com.vividsolutions.jts.geom.Geometry surfaceToJts(Geometry shp, int srid){
    com.vividsolutions.jts.geom.Geometry geoJTS = null;
    ArrayList arrayLines;
    LineString lin = null;
    PathIterator theIterator;
    int theType;
    int numParts = 0;
    double[] theData = new double[6];
    ArrayList arrayCoords = null;
    Coordinate coord;

    arrayLines = new ArrayList();

    ArrayList shells = new ArrayList();
    ArrayList holes = new ArrayList();
    Coordinate[] points = null;

    theIterator = shp.getPathIterator(null, manager.getFlatness());

    while (!theIterator.isDone()) {
        //while not done
        theType = theIterator.currentSegment(theData);

        //Populate a segment of the new
        // GeneralPathX object.
        //Process the current segment to populate a new
        // segment of the new GeneralPathX object.
        switch (theType) {
        case PathIterator.SEG_MOVETO:

            // System.out.println("SEG_MOVETO");
            if (arrayCoords == null) {
                arrayCoords = new ArrayList();
            } else {
                points = CoordinateArrays.toCoordinateArray(arrayCoords);

                try {
                    LinearRing ring = geomFactory.createLinearRing(points);

                    if (CGAlgorithms.isCCW(points)) {
                        holes.add(ring);
                    } else {
                        shells.add(ring);
                    }
                } catch (Exception e) {
                    /* (jaume) caso cuando todos los puntos son iguales
                     * devuelvo el propio punto
                     */
                    boolean same = true;
                    for (int i = 0; i < points.length-1 && same; i++) {
                        if (points[i].x != points[i+1].x ||
                                points[i].y != points[i+1].y /*||
                                points[i].z != points[i+1].z*/
```

```
                ) {
                    same = false;
                }
            }
            if (same) {
                return geomFactory.createPoint(points[0]);
            }
            /*
             * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
             * una linea
             */
            if (points.length>1 && points.length<=3) {
                // return geomFactory.createLineString(points);
                return geomFactory.createMultiLineString(new LineString[] {geomFactory.createLineString(points)});
            }

            System.err.println(
            "Caught Topology exception in GMLLinearRingHandler");

            return null;
        }

        /* if (numParts == 1)
         {
         linRingExt = new GeometryFactory().createLinearRing(
         CoordinateArrays.toCoordinateArray(arrayCoords));
         }
         else
         {
         linRing = new GeometryFactory().createLinearRing(
         CoordinateArrays.toCoordinateArray(arrayCoords));
         arrayLines.add(linRing);
         } */
        arrayCoords = new ArrayList();
    }

    numParts++;
    arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

    break;

case PathIterator.SEG_LINETO:

    // System.out.println("SEG_LINETO");
    arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

    break;

case PathIterator.SEG_QUADTO:
    System.out.println("SEG_QUADTO Not supported here");
```

```
            break;

        case PathIterator.SEG_CUBICTO:
            System.out.println("SEG_CUBICTO Not supported here");

            break;

        case PathIterator.SEG_CLOSE:

            // Aï¿½adimos el primer punto para cerrar.
            Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
            arrayCoords.add(new Coordinate(firstCoord.x,
                    firstCoord.y));

            break;
        } //end switch

        // System.out.println("theData[0] = " + theData[0] + " theData[1]=" + theData[1]);
        theIterator.next();
    } //end while loop

    Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
    Coordinate lastCoord = (Coordinate) arrayCoords.get(arrayCoords
            .size() - 1);
    if (!isClosed(firstCoord, lastCoord)) {
        arrayCoords.add(firstCoord);
    }
    points = CoordinateArrays.toCoordinateArray(arrayCoords);

    try {
        LinearRing ring = geomFactory.createLinearRing(points);

        if (CGAlgorithms.isCCW(points)) {
            holes.add(ring);
        } else {
            shells.add(ring);
        }
        ring.setSRID(srid);
    } catch (Exception e) {
        /* (jaume) caso cuando todos los puntos son iguales
         * devuelvo el propio punto
         */
        boolean same = true;
        for (int i = 0; i < points.length-1 && same; i++) {
            if (points[i].x != points[i+1].x ||
                    points[i].y != points[i+1].y /*||
                    points[i].z != points[i+1].z*/
            ) {
                same = false;
            }
        }
        if (same) {
            geoJTS = geomFactory.createPoint(points[0]);
```

```
            geoJTS.setSRID(srid);
            return geoJTS;
        }
        /*
         * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
         * una linea
         */
        if (points.length>1 && points.length<=3) {
            // return geomFactory.createLineString(points);
            geoJTS = geomFactory
            .createMultiLineString(new LineString[] { geomFactory
                    .createLineString(points) });
            geoJTS.setSRID(srid);
            return geoJTS;
        }
        System.err.println(
        "Caught Topology exception in GMLLinearRingHandler");

        return null;
    }

    /* linRing = new GeometryFactory().createLinearRing(
     CoordinateArrays.toCoordinateArray(arrayCoords)); */

    // System.out.println("NumParts = " + numParts);
    //now we have a list of all shells and all holes
    ArrayList holesForShells = new ArrayList(shells.size());

    for (int i = 0; i < shells.size(); i++) {
        holesForShells.add(new ArrayList());
    }

    //find homes
    for (int i = 0; i < holes.size(); i++) {
        LinearRing testRing = (LinearRing) holes.get(i);
        LinearRing minShell = null;
        Envelope minEnv = null;
        Envelope testEnv = testRing.getEnvelopeInternal();
        Coordinate testPt = testRing.getCoordinateN(0);
        LinearRing tryRing = null;

        for (int j = 0; j < shells.size(); j++) {
            tryRing = (LinearRing) shells.get(j);

            Envelope tryEnv = tryRing.getEnvelopeInternal();

            if (minShell != null) {
                minEnv = minShell.getEnvelopeInternal();
            }

            boolean isContained = false;
            Coordinate[] coordList = tryRing.getCoordinates();
```

```
        if (tryEnv.contains(testEnv) &&
              (CGAlgorithms.isPointInRing(testPt, coordList) ||
                    (pointInList(testPt, coordList)))) {
           isContained = true;
        }

        // check if this new containing ring is smaller than the current minimum ring
        if (isContained) {
           if ((minShell == null) || minEnv.contains(tryEnv)) {
              minShell = tryRing;
           }
        }
     }

     if (minShell == null) {
        //                System.out.println(
        //                "polygon found with a hole thats not inside a shell");
        //                azabala: we do the assumption that this hole is really a shell (polygon)
        //                whose point werent digitized in the right order
        Coordinate[] cs = testRing.getCoordinates();
        Coordinate[] reversed = new Coordinate[cs.length];
        int pointIndex = 0;
        for(int z = cs.length-1; z >= 0; z--){
           reversed[pointIndex] = cs[z];
           pointIndex++;
        }
        LinearRing newRing = geomFactory.createLinearRing(reversed);
        shells.add(newRing);
        holesForShells.add(new ArrayList());
     } else {
        ((ArrayList) holesForShells.get(shells.indexOf(minShell))).add(testRing);
     }
  }

  Polygon[] polygons = new Polygon[shells.size()];

  for (int i = 0; i < shells.size(); i++) {
     polygons[i] = geomFactory.createPolygon((LinearRing) shells.get(
           i),
           (LinearRing[]) ((ArrayList) holesForShells.get(i)).toArray(
                 new LinearRing[0]));
     polygons[i].setSRID(srid);
  }

  holesForShells = null;
  shells = null;
  holes = null;

  geoJTS = geomFactory.createMultiPolygon(polygons);
  geoJTS.setSRID(srid);

  return geoJTS;
```

```
        }
```

<Spanish>
Se han detectado algunos fallos en la clase Converter cuando trata con multipoint, multisurfaces y multicurve (o no lo tiene en cuenta o se produce un fallo)

A continuación se indica una propuesta con las correciones a hacer. Este código está probado y se está usando en una aplicación ahora mismo en funcionamiento (gisEIEL 2.3) y otra a punto de ser publicada (gisEIEL 3.0 ésta última basada en gvSIG 2.0)

- En el método jtsToGeometry falta el caso de que la geometría sea de tipo MultiPoint (entre * se indica el código a añadir)

```
        if (geo.isEmpty()) {
            shpNew = null;
        }

        try{
            *if (geo instanceof com.vividsolutions.jts.geom.MultiPoint) {
                shpNew = geomManager.create(TYPES.MULTIPOINT, SUBTYPES.GEOM2D);
                for (int i = 0; i < geo.getNumGeometries(); i++) {
                    Point point = (Point) geo.getGeometryN(i);
                    ((MultiPoint)shpNew).addPoint((org.gvsig.fmap.geom.primitive.Point) Converter.jtsToGeometry(point));
                }

            }*

            if (geo instanceof Polygon) {
                shpNew = geomManager.createSurface(toShape((Polygon) geo), SUBTYPES.GEOM2D);
            }

            ....
```

- Cambios en el método *com.vividsolutions.jts.geom.Geometry geometryToJts(Geometry shp, int shapeType, int srid)* para redefinir los casos de multipoint, multisurface y multicurve y para arreglar un fallo en el caso de ellipse
ELLIPSE

```
    case Geometry.TYPES.ELLIPSE:
            arrayLines = new ArrayList();

            ArrayList shells = new ArrayList();
            ArrayList holes = new ArrayList();
            Coordinate[] points = null;

            theIterator = shp.getPathIterator(null, manager.getFlatness());

            while (!theIterator.isDone()) {
                //while not done
                theType = theIterator.currentSegment(theData);

                //Populate a segment of the new
                // GeneralPathX object.
```

```
//Process the current segment to populate a new
// segment of the new GeneralPathX object.
switch (theType) {
case PathIterator.SEG_MOVETO:

   // System.out.println("SEG_MOVETO");
   if (arrayCoords == null) {
      arrayCoords = new ArrayList();
   } else {
      points = CoordinateArrays.toCoordinateArray(arrayCoords);

      try {
         LinearRing ring = geomFactory.createLinearRing(points);

         if (CGAlgorithms.isCCW(points)) {
            holes.add(ring);
         } else {
            shells.add(ring);
         }
      } catch (Exception e) {
         /* (jaume) caso cuando todos los puntos son iguales
          * devuelvo el propio punto
          */
         boolean same = true;
         for (int i = 0; i < points.length-1 && same; i++) {
            if (points[i].x != points[i+1].x ||
                  points[i].y != points[i+1].y /*||
                  points[i].z != points[i+1].z*/
            ) {
               same = false;
            }
         }
         if (same) {
            return geomFactory.createPoint(points[0]);
         }
         /*
          * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
          * una linea
          */
         if (points.length>1 && points.length<=3) {
            // return geomFactory.createLineString(points);
            return geomFactory.createMultiLineString(new LineString[] {geomFactory.createLineString(points)});
         }

         System.err.println(
         "Caught Topology exception in GMLLinearRingHandler");

         return null;
      }

      /* if (numParts == 1)
       {
       linRingExt = new GeometryFactory().createLinearRing(
```

```
                CoordinateArrays.toCoordinateArray(arrayCoords));
                }
                else
                {
                linRing = new GeometryFactory().createLinearRing(
                CoordinateArrays.toCoordinateArray(arrayCoords));
                arrayLines.add(linRing);
                } */
               arrayCoords = new ArrayList();
            }

            numParts++;
            arrayCoords.add(new Coordinate(theData[0],
                  theData[1]));

            break;

         case PathIterator.SEG_LINETO:

            // System.out.println("SEG_LINETO");
            arrayCoords.add(new Coordinate(theData[0],
                  theData[1]));

            break;

         case PathIterator.SEG_QUADTO:
            System.out.println("SEG_QUADTO Not supported here");

            break;

         case PathIterator.SEG_CUBICTO:
            System.out.println("SEG_CUBICTO Not supported here");

            break;

         case PathIterator.SEG_CLOSE:

            // Aï¿½adimos el primer punto para cerrar.
            Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
            arrayCoords.add(new Coordinate(firstCoord.x,
                  firstCoord.y));

            break;
         } //end switch

         // System.out.println("theData[0] = " + theData[0] + " theData[1]=" + theData[1]);
         theIterator.next();
      } //end while loop

      Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
      Coordinate lastCoord = (Coordinate) arrayCoords.get(arrayCoords
            .size() - 1);
      if (!isClosed(firstCoord, lastCoord)) {
```

```
            arrayCoords.add(firstCoord);
        }
        points = CoordinateArrays.toCoordinateArray(arrayCoords);

        try {
            LinearRing ring = geomFactory.createLinearRing(points);

            if (CGAlgorithms.isCCW(points)) {
                holes.add(ring);
            } else {
                shells.add(ring);
            }
            ring.setSRID(srid);
        } catch (Exception e) {
            /* (jaume) caso cuando todos los puntos son iguales
             * devuelvo el propio punto
             */
            boolean same = true;
            for (int i = 0; i < points.length-1 && same; i++) {
                if (points[i].x != points[i+1].x ||
                        points[i].y != points[i+1].y /*||
                        points[i].z != points[i+1].z*/
                ) {
                    same = false;
                }
            }
            if (same) {
                geoJTS = geomFactory.createPoint(points[0]);
                geoJTS.setSRID(srid);
                return geoJTS;
            }
            /*
             * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
             * una linea
             */
            if (points.length>1 && points.length<=3) {
                // return geomFactory.createLineString(points);
                geoJTS = geomFactory
                .createMultiLineString(new LineString[] { geomFactory
                        .createLineString(points) });
                geoJTS.setSRID(srid);
                return geoJTS;
            }
            System.err.println(
            "Caught Topology exception in GMLLinearRingHandler");

            return null;
        }

        /* linRing = new GeometryFactory().createLinearRing(
         CoordinateArrays.toCoordinateArray(arrayCoords)); */

        // System.out.println("NumParts = " + numParts);
```

```
//now we have a list of all shells and all holes
ArrayList holesForShells = new ArrayList(shells.size());

for (int i = 0; i < shells.size(); i++) {
    holesForShells.add(new ArrayList());
}

//find homes
for (int i = 0; i < holes.size(); i++) {
    LinearRing testRing = (LinearRing) holes.get(i);
    LinearRing minShell = null;
    Envelope minEnv = null;
    Envelope testEnv = testRing.getEnvelopeInternal();
    Coordinate testPt = testRing.getCoordinateN(0);
    LinearRing tryRing = null;

    for (int j = 0; j < shells.size(); j++) {
        tryRing = (LinearRing) shells.get(j);

        Envelope tryEnv = tryRing.getEnvelopeInternal();

        if (minShell != null) {
            minEnv = minShell.getEnvelopeInternal();
        }

        boolean isContained = false;
        Coordinate[] coordList = tryRing.getCoordinates();

        if (tryEnv.contains(testEnv) &&
                (CGAlgorithms.isPointInRing(testPt, coordList) ||
                        (pointInList(testPt, coordList)))) {
            isContained = true;
        }

        // check if this new containing ring is smaller than the current minimum ring
        if (isContained) {
            if ((minShell == null) || minEnv.contains(tryEnv)) {
                minShell = tryRing;
            }
        }
    }

    if (minShell == null) {
        //                  System.out.println(
        //                  "polygon found with a hole thats not inside a shell");
        //                  azabala: we do the assumption that this hole is really a shell (polygon)
        //                  whose point werent digitized in the right order
        Coordinate[] cs = testRing.getCoordinates();
        Coordinate[] reversed = new Coordinate[cs.length];
        int pointIndex = 0;
        for(int z = cs.length-1; z >= 0; z--){
            reversed[pointIndex] = cs[z];
            pointIndex++;
```

```
        }
        LinearRing newRing = geomFactory.createLinearRing(reversed);
        shells.add(newRing);
        holesForShells.add(new ArrayList());
    } else {
        ((ArrayList) holesForShells.get(shells.indexOf(minShell))).add(testRing);
    }
}

Polygon[] polygons = new Polygon[shells.size()];

for (int i = 0; i < shells.size(); i++) {
    polygons[i] = geomFactory.createPolygon((LinearRing) shells.get(
            i),
            (LinearRing[]) ((ArrayList) holesForShells.get(i)).toArray(
                    new LinearRing[0]));
    polygons[i].setSRID(srid);
}
// CAMBIO: ENTREGAMOS SIEMPRE MULTILINESTRING, QUE ES
// LO QUE HACE TODO EL MUNDO CUANDO ESCRIBE EN POSTGIS
// O CON GEOTOOLS
// if (numParts > 1) // Generamos una MultiLineString

/* if (polygons.length == 1) {
 return polygons[0];
 } */

// FIN CAMBIO

holesForShells = null;
shells = null;
holes = null;

/*if (polygons.length == 1) {
    geoJTS = polygons[0];
} else {
    // its a multi part
    geoJTS = geomFactory.createMultiPolygon(polygons);
}
geoJTS.setSRID(srid);*/
//its a multi part
geoJTS = geomFactory.createMultiPolygon(polygons);
geoJTS.setSRID(srid);

/* if (numParts > 1) // Generamos un Polygon con agujeros
 {
 arrayLines.add(linRing);
 // geoJTS = new GeometryFactory().createPolygon(linRingExt,
  // GeometryFactory.toLinearRingArray(arrayLines));
   geoJTS = new GeometryFactory().buildGeometry(arrayLines);

   // geoJTS = Polygonizer.class.
    }
```

```
                else
                {
                geoJTS = new GeometryFactory().createPolygon(linRing,null);
                } */
            break;
```

MULTICURVE, MULTIPOINT, MULTISURFACE

```
    case Geometry.TYPES.MULTICURVE:
            geoJTS = multiCurveToJts((MultiCurve)shp, srid);
            geoJTS.setSRID(srid);
            break;

        case Geometry.TYPES.MULTIPOINT:
            geoJTS = geometryToJts((MultiPoint)shp);
            geoJTS.setSRID(srid);
            break;

        case Geometry.TYPES.MULTISURFACE:
            geoJTS = multiSurfaceToJts((MultiSurface)shp, srid);
            geoJTS.setSRID(srid);
            break;
```

- Creación de métodos nuevos para soportar los cambios anteriores

```
    public static com.vividsolutions.jts.geom.Geometry multiCurveToJts(MultiCurve geom, int srid) {
        LineString[] lines = new LineString[geom.getPrimitivesNumber()];
        for (int i = 0; i < lines.length; i++){
            lines[i] = (LineString) curveToJts((geom.getPrimitiveAt(i)), srid);
        }
        return new com.vividsolutions.jts.geom.GeometryFactory().createMultiLineString(lines);
    }

    private static com.vividsolutions.jts.geom.Geometry curveToJts(Geometry shp, int srid){

        ArrayList arrayLines;
        LineString lin = null;
        PathIterator theIterator;
        int theType;
        int numParts = 0;
        double[] theData = new double[6];
        ArrayList arrayCoords = null;
        Coordinate coord;

        arrayLines = new ArrayList();
        theIterator = shp.getPathIterator(null, manager.getFlatness());

        while (!theIterator.isDone()) {
            //while not done
```

```
theType = theIterator.currentSegment(theData);

//Populate a segment of the new
// GeneralPathX object.
//Process the current segment to populate a new
// segment of the new GeneralPathX object.
switch (theType) {
case PathIterator.SEG_MOVETO:

   // System.out.println("SEG_MOVETO");
   if (arrayCoords == null) {
      arrayCoords = new ArrayList();
   } else {
      lin = geomFactory.createLineString(CoordinateArrays.toCoordinateArray(
            arrayCoords));
      lin.setSRID(srid);
      arrayLines.add(lin);
      arrayCoords = new ArrayList();
   }

   numParts++;
   coord = new Coordinate(theData[0], theData[1]);

   arrayCoords.add(coord);

   break;

case PathIterator.SEG_LINETO:

   // System.out.println("SEG_LINETO");
   arrayCoords.add(new Coordinate(theData[0],
         theData[1]));

   break;

case PathIterator.SEG_QUADTO:
   System.out.println("Not supported here");

   break;

case PathIterator.SEG_CUBICTO:
   System.out.println("Not supported here");

   break;

case PathIterator.SEG_CLOSE:
   // Aï¿½adimos el primer punto para cerrar.
   Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
   // Solo anyadimos cuando no esta ya cerrado
   arrayCoords.add(new Coordinate(firstCoord.x,
         firstCoord.y));

   break;
```

```
            } //end switch

            theIterator.next();
        } //end while loop

        if (arrayCoords.size()<2) {

        }else{

            lin = new com.vividsolutions.jts.geom.GeometryFactory().createLineString(CoordinateArrays.toCoordinateArray(
                    arrayCoords));

            lin.setSRID(srid);

        }

        return lin;

    }



/**
     * Método creado para construir un MultiSurface formado por varias surface
     *
     * @author Leticia Riestra
     * @param geom
     * @return
     */
    public static com.vividsolutions.jts.geom.Geometry multiSurfaceToJts(MultiSurface geom, int srid) {
        Polygon[] polygons = new Polygon[geom.getPrimitivesNumber()];
        for (int i = 0; i < polygons.length; i++){
            MultiPolygon polygon = (MultiPolygon)surfaceToJts((geom.getPrimitiveAt(i)), srid);

            polygons[i] = (Polygon)polygon.getGeometryN(0);//(Polygon) surfaceToJts((geom.getPrimitiveAt(i)), srid);
        }
        return new com.vividsolutions.jts.geom.GeometryFactory().createMultiPolygon(polygons);
    }

    private static com.vividsolutions.jts.geom.Geometry surfaceToJts(Geometry shp, int srid){
        com.vividsolutions.jts.geom.Geometry geoJTS = null;
        ArrayList arrayLines;
        LineString lin = null;
        PathIterator theIterator;
        int theType;
        int numParts = 0;
        double[] theData = new double[6];
        ArrayList arrayCoords = null;
        Coordinate coord;

        arrayLines = new ArrayList();

        ArrayList shells = new ArrayList();
```

```
ArrayList holes = new ArrayList();
Coordinate[] points = null;

theIterator = shp.getPathIterator(null, manager.getFlatness());

while (!theIterator.isDone()) {
   //while not done
   theType = theIterator.currentSegment(theData);

   //Populate a segment of the new
   // GeneralPathX object.
   //Process the current segment to populate a new
   // segment of the new GeneralPathX object.
   switch (theType) {
   case PathIterator.SEG_MOVETO:

      // System.out.println("SEG_MOVETO");
      if (arrayCoords == null) {
         arrayCoords = new ArrayList();
      } else {
         points = CoordinateArrays.toCoordinateArray(arrayCoords);

         try {
            LinearRing ring = geomFactory.createLinearRing(points);

            if (CGAlgorithms.isCCW(points)) {
               holes.add(ring);
            } else {
               shells.add(ring);
            }
         } catch (Exception e) {
            /* (jaume) caso cuando todos los puntos son iguales
             * devuelvo el propio punto
             */
            boolean same = true;
            for (int i = 0; i < points.length-1 && same; i++) {
               if (points[i].x != points[i+1].x ||
                     points[i].y != points[i+1].y /*||
                     points[i].z != points[i+1].z*/
               ) {
                  same = false;
               }
            }
            if (same) {
               return geomFactory.createPoint(points[0]);
            }
            /*
             * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
             * una linea
             */
            if (points.length>1 && points.length<=3) {
               // return geomFactory.createLineString(points);
               return geomFactory.createMultiLineString(new LineString[] {geomFactory.createLineString(points)});
```

```
            }

            System.err.println(
            "Caught Topology exception in GMLLinearRingHandler");

            return null;
          }

          /* if (numParts == 1)
           {
           linRingExt = new GeometryFactory().createLinearRing(
           CoordinateArrays.toCoordinateArray(arrayCoords));
           }
           else
           {
           linRing = new GeometryFactory().createLinearRing(
           CoordinateArrays.toCoordinateArray(arrayCoords));
           arrayLines.add(linRing);
           } */
          arrayCoords = new ArrayList();
      }

      numParts++;
      arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

      break;

  case PathIterator.SEG_LINETO:

      // System.out.println("SEG_LINETO");
      arrayCoords.add(new Coordinate(theData[0],
            theData[1]));

      break;

  case PathIterator.SEG_QUADTO:
      System.out.println("SEG_QUADTO Not supported here");

      break;

  case PathIterator.SEG_CUBICTO:
      System.out.println("SEG_CUBICTO Not supported here");

      break;

  case PathIterator.SEG_CLOSE:

      // Aï¿½adimos el primer punto para cerrar.
      Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
      arrayCoords.add(new Coordinate(firstCoord.x,
            firstCoord.y));
```

```
            break;
        } //end switch

        // System.out.println("theData[0] = " + theData[0] + " theData[1]=" + theData[1]);
        theIterator.next();
    } //end while loop

    Coordinate firstCoord = (Coordinate) arrayCoords.get(0);
    Coordinate lastCoord = (Coordinate) arrayCoords.get(arrayCoords
            .size() - 1);
    if (!isClosed(firstCoord, lastCoord)) {
        arrayCoords.add(firstCoord);
    }
    points = CoordinateArrays.toCoordinateArray(arrayCoords);

    try {
        LinearRing ring = geomFactory.createLinearRing(points);

        if (CGAlgorithms.isCCW(points)) {
            holes.add(ring);
        } else {
            shells.add(ring);
        }
        ring.setSRID(srid);
    } catch (Exception e) {
        /* (jaume) caso cuando todos los puntos son iguales
         * devuelvo el propio punto
         */
        boolean same = true;
        for (int i = 0; i < points.length-1 && same; i++) {
            if (points[i].x != points[i+1].x ||
                    points[i].y != points[i+1].y /*||
                    points[i].z != points[i+1].z*/
            ) {
                same = false;
            }
        }
        if (same) {
            geoJTS = geomFactory.createPoint(points[0]);
            geoJTS.setSRID(srid);
            return geoJTS;
        }
        /*
         * caso cuando es una lï¿½nea de 3 puntos, no creo un LinearRing, sino
         * una linea
         */
        if (points.length>1 && points.length<=3) {
            // return geomFactory.createLineString(points);
            geoJTS = geomFactory
            .createMultiLineString(new LineString[] { geomFactory
                    .createLineString(points) });
            geoJTS.setSRID(srid);
            return geoJTS;
```

```
    }
    System.err.println(
    "Caught Topology exception in GMLLinearRingHandler");

    return null;
}

/* linRing = new GeometryFactory().createLinearRing(
 CoordinateArrays.toCoordinateArray(arrayCoords)); */

// System.out.println("NumParts = " + numParts);
//now we have a list of all shells and all holes
ArrayList holesForShells = new ArrayList(shells.size());

for (int i = 0; i < shells.size(); i++) {
    holesForShells.add(new ArrayList());
}

//find homes
for (int i = 0; i < holes.size(); i++) {
    LinearRing testRing = (LinearRing) holes.get(i);
    LinearRing minShell = null;
    Envelope minEnv = null;
    Envelope testEnv = testRing.getEnvelopeInternal();
    Coordinate testPt = testRing.getCoordinateN(0);
    LinearRing tryRing = null;

    for (int j = 0; j < shells.size(); j++) {
        tryRing = (LinearRing) shells.get(j);

        Envelope tryEnv = tryRing.getEnvelopeInternal();

        if (minShell != null) {
            minEnv = minShell.getEnvelopeInternal();
        }

        boolean isContained = false;
        Coordinate[] coordList = tryRing.getCoordinates();

        if (tryEnv.contains(testEnv) &&
                (CGAlgorithms.isPointInRing(testPt, coordList) ||
                        (pointInList(testPt, coordList)))) {
            isContained = true;
        }

        // check if this new containing ring is smaller than the current minimum ring
        if (isContained) {
            if ((minShell == null) || minEnv.contains(tryEnv)) {
                minShell = tryRing;
            }
        }
    }
```

```
            if (minShell == null) {
                //                  System.out.println(
                //                  "polygon found with a hole thats not inside a shell");
                //                  azabala: we do the assumption that this hole is really a shell (polygon)
                //                  whose point werent digitized in the right order
                Coordinate[] cs = testRing.getCoordinates();
                Coordinate[] reversed = new Coordinate[cs.length];
                int pointIndex = 0;
                for(int z = cs.length-1; z >= 0; z--){
                    reversed[pointIndex] = cs[z];
                    pointIndex++;
                }
                LinearRing newRing = geomFactory.createLinearRing(reversed);
                shells.add(newRing);
                holesForShells.add(new ArrayList());
            } else {
                ((ArrayList) holesForShells.get(shells.indexOf(minShell))).add(testRing);
            }
        }

        Polygon[] polygons = new Polygon[shells.size()];

        for (int i = 0; i < shells.size(); i++) {
            polygons[i] = geomFactory.createPolygon((LinearRing) shells.get(
                    i),
                    (LinearRing[]) ((ArrayList) holesForShells.get(i)).toArray(
                            new LinearRing[0]));
            polygons[i].setSRID(srid);
        }

        holesForShells = null;
        shells = null;
        holes = null;

        geoJTS = geomFactory.createMultiPolygon(polygons);
        geoJTS.setSRID(srid);

        return geoJTS;

    }
```

Associated revisions

Revision 38392 - 06/08/2012 08:39 AM - Ignacio Brodín

fixes #546 Error in class Converter with one solution. Patch from Leticia Riestra

History

#1 - 04/19/2012 01:38 PM - Joaquín del Cerro Murciano

Hola Leticia,

he estado intentando ver de aplicar los cambios que me has comentado a ver en que consistian y no se si es que me he liado pero no he logrado hacerlo. Se me quejaba de un par de errores, de un "case" duplicado

```
case Geometry.TYPES.MULTIPOINT:
```

Y de que trataba de invocar a un metodo que no era estatico (me suena que ya habias comentado algo de eso).

De rebote, mirando tus cambios, he visto otro error que arreglare despues de ver meter tus cambios para que no se mezclen.

Bueno, a lo que iba, que si me puedes adjuntar al ticket la clase Converter completa para que al hacer el merge no me deje nada casi que mejor, si no ire tocando cosas y igual me dejo alguna.

Y muchas gracias por el parche.

Un saludo
Joaquin

#2 - 04/19/2012 02:14 PM - Leticia Riestra

- File Converter.java added

Hola

Te envío en adjunto la clase Converter

Me alegro que te haya servido. Todo lo que sea colaborar con vosotros, nosotros encantados

Saludos

#3 - 04/19/2012 02:35 PM - Joaquín del Cerro Murciano

De los cambios que he visto has pasado hay uno que, desde mi desconocimiento del manejo de geometrias, me preocupa un poco.

Cuando se convertia una geometria gvSIG de tipo Surface a JTS si el resultado era un solo poligono, se devolvia un Polygon de jts, si eran varios se devolvia un MultiPolygon de jts. Con el cambio que propones siempre se devuelve un MultiPolygon aunque tengas un solo poligono. No tengo ni idea de las repercusiones de eso.

Si puedes comentar un poco por que ese cambio para ver si lo entiendo mejor te lo agradeceria.

(ahora le hecho un vistazo al adjunto)

Un saludo
Joaquin

#4 - 04/19/2012 04:00 PM - Jorge Sanz

Leticia Riestra wrote:

> *Hola*
>
> *Te envío en adjunto la clase Converter*
>
> *Me alegro que te haya servido. Todo lo que sea colaborar con vosotros,*
> *nosotros encantados*
>
> *Saludos*

Leticia, para aceptar los cambios a componentes del core necesitamos que nos envieis la CLA del proyecto. Este procedimiento asegura que el copyright del código de gvSIG está a cargo de una única entidad (la Asociación gvSIG) manteniendo por supuesto la atribución a vuestro trabajo.

En http://is.gd/TmV7KI se puede consultar qué es la CLA, hay enlaces al documento para rellenarlo y enviarlo a la Asociación.

Por lo demás por supuesto quedamos a vuestra disposición para resolver cualquier duda.

Un saludo y muchas gracias por colaborar!

#5 - 04/19/2012 05:08 PM - Leticia Riestra

Hola

Con respecto al CLA os lo hemos intentando enviar ahora por FAX pero nos ha dado un error. Os lo enviamos por pdf

Con respecto a lo de la clase Converter, si te soy sincera, no me acuerdo porque llevamos haciendo cosas con vuestro código desde 2009-2010 y ésto fue una de las primera cosas que revisamos y ahora mismo no me acuerdo el motivo de que lo hicieramos.
Creo recordar (y digo creo) que el motivo era porque al final trataba igual los multipolygon de los polygon. Ya te digo que nosotros lo estamos usando tal y como te lo he enviado y cargamos capas multipolygon y polygon a secas y funciona perfectamente

De todas formas, si lo crees conveniente, puedes poner ese código como estaba (si te fijas lo tengo comentado justo en la líne de arriba) y comprobar que funciona

Saludos

#6 - 04/21/2012 11:07 PM - Joaquín del Cerro Murciano

Leticia Riestra wrote:

> *Hola*
>
> *Te envío en adjunto la clase Converter*
>
> *Me alegro que te haya servido. Todo lo que sea colaborar con vosotros,*
> *nosotros encantados*

Hola.

Despues de echarle un vistazo rapido, aparentemente parece todo bien
con excepcion de:

- Lo de que antes habia un caso que devolvia Polygon y ahora MultiPoligon.
- En el metodo geometryToJts(Geometry shp, int shapeType, int srid) ha desaparecido el tratamiento de Geometry.TYPES.SPLINE, no se si se ha quitado por alguna razon especial.

Si sabes algo de lo del SPLINE que me pueda aportar alguna pista mejor, si no en principio veriamos de subir los cambios tuyos añadiendo el SPLINE.

Un saludo
Joaquin

> *Saludos*

#7 - 04/23/2012 10:03 AM - Leticia Riestra

Hola
Con respecto al método spline tienes razón. Se me escapó sin querer, así que lo puedes añadir sin problemas. Perdona

#8 - 04/24/2012 03:25 PM - Jorge Sanz

Leticia Riestra wrote:

> *Hola*
>
> *Con respecto al CLA os lo hemos intentando enviar ahora por FAX pero nos ha dado un error. Os lo enviamos por pdf*
>
> *Con respecto a lo de la clase Converter, si te soy sincera, no me acuerdo porque llevamos haciendo cosas con vuestro código desde 2009-2010 y ésto fue una de las primera cosas que revisamos y ahora mismo no me acuerdo el motivo de que lo hicieramos.*
> *Creo recordar (y digo creo) que el motivo era porque al final trataba igual los multipolygon de los polygon. Ya te digo que nosotros lo estamos usando tal y como te lo he enviado y cargamos capas multipolygon y polygon a secas y funciona perfectamente*
>
> *De todas formas, si lo crees conveniente, puedes poner ese código como estaba (si te fijas lo tengo comentado justo en la líne de arriba) y comprobar que funciona*
>
> *Saludos*

Buenas, solo informar de que hemos recibido y archivado la CLA del laboratorio (falta que la firme Álvaro pero está de viaje).

El laboratorio ya aparece en el listado de organizaciones que ha facilitado la CLA.

https://gvsig.org/web/projects/gvsig-desktop/docs/devel/como-contribuir-en-gvsig/contribuciones-y-parches-al-codigo-de-gvsig/cla-status

Muchas gracias por hacer fácil lo que a otros les parece tan difícil!!
Jorge

#9 - 04/30/2012 10:49 AM - Joaquín del Cerro Murciano

- Assignee set to Joaquín del Cerro Murciano

- Has patch changed from No to Yes

#10 - 05/10/2012 02:44 PM - Manuel Madrid

- Target version set to 2.0.0-rc1

#11 - 06/08/2012 02:40 PM - Ignacio Brodín

- Status changed from New to Fixed

- % Done changed from 0 to 100

Applied in changeset r38392.

#12 - 06/11/2012 01:28 PM - Joaquín del Cerro Murciano

- Target version changed from 2.0.0-rc1 to 2.0.0-devel-2048

#13 - 08/29/2012 02:25 PM - Joaquín del Cerro Murciano

- Status changed from Fixed to Closed

Files

Converter.java	47.7 KB	04/19/2012	Leticia Riestra