Scientific Visualization: Übung 3

von
Naja v. Schmude und Lisa Dohrmann

Aufgabe 6: Eulercharakteristik in JavaView

Die relevanten Codeteile stehen in `PwEuler.java`.

```
/**
 * Computes the Euler characteristic for the the given
 * geometry.
 */
public int computeEulerCharacteristic(PgElementSet elementSet
    ) {
  int[] fvector = computeFVector(elementSet);
  int euler = 0;

  PsDebug.message("Geometry: " +elementSet.getName());
  PsDebug.message(
      "\tvertices = " + fvector[0] + "\n" +
      "\tedges = " + fvector[1] + "\n" +
      "\tfaces = " + fvector[2]);

  for (int i = 0; i < fvector.length; ++i) {
    euler += Math.pow(-1, i) * fvector[i];
  }

  PsDebug.message("Euler characteristic = " + euler);
  return euler;
}

/**
 * Computes the F-Vector for the given geometry by counting
 * the vertices and faces and computing the edges.
 * <br>
 * NOTE: A triangulated geometry is a prerequisite.
 */
public static int[] computeFVector(PgElementSet elementSet) {
  if (elementSet == null)
    throw new IllegalArgumentException("Geometry must not be
        null.");

  int vertices, edges, faces = 0;
  PiVector[] elements = elementSet.getElements();
  Set<Integer> vertexSet = new HashSet<Integer>();

  for (PiVector vec : elements) {
    // Test for null objects, because the documentation says
    // that the elements array might be larger than the
    // number of elements it contains
    if (vec == null) continue;

    // We count the faces by ourselves because
    // elements.length might be inaccurate (see above)
    if (vec.getSize() == 3) faces++;

    // collect all vertices in a set
    for (int i = 0; i < vec.getSize(); i++) {
      vertexSet.add(vec.getEntry(i));
```

```
      }
    }

    vertices = vertexSet.size();
    // compute the edges from the number of triangles
    edges = (int) Math.floor(3.0 / 2.0 * faces);

    return new int[] {vertices, edges, faces};
  }
```