Geometric Modeling

2015

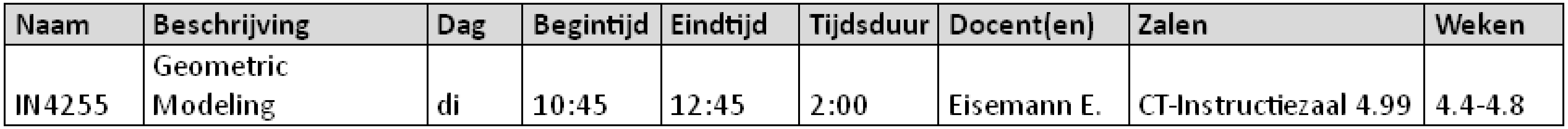
Tutorial: Introduction to JavaView

TUDelft

Announcement

New Lecture Room

- The lectures will be in room CT4.99 (Civil Engineering)



Naam	Beschrijving	Dag	Begintijd	Eindtijd	Tijdsduur	Docent(en)	Zalen	Weken
IN4255	Geometric Modeling	di	10:45	12:45	2:00	Eisemann E.	CT-Instructiezaal 4.99	4.4-4.8

Assignements

Theoretical Assignment 1

- No hand-in
- Will be discussed in the tutorial on May, 20th

Practical Assignment 1

- Due: 26-05-2015
- JavaView framework is provided: javaview.zip, models.zip in Blackboard
- Self enroll in groups in Blackboard

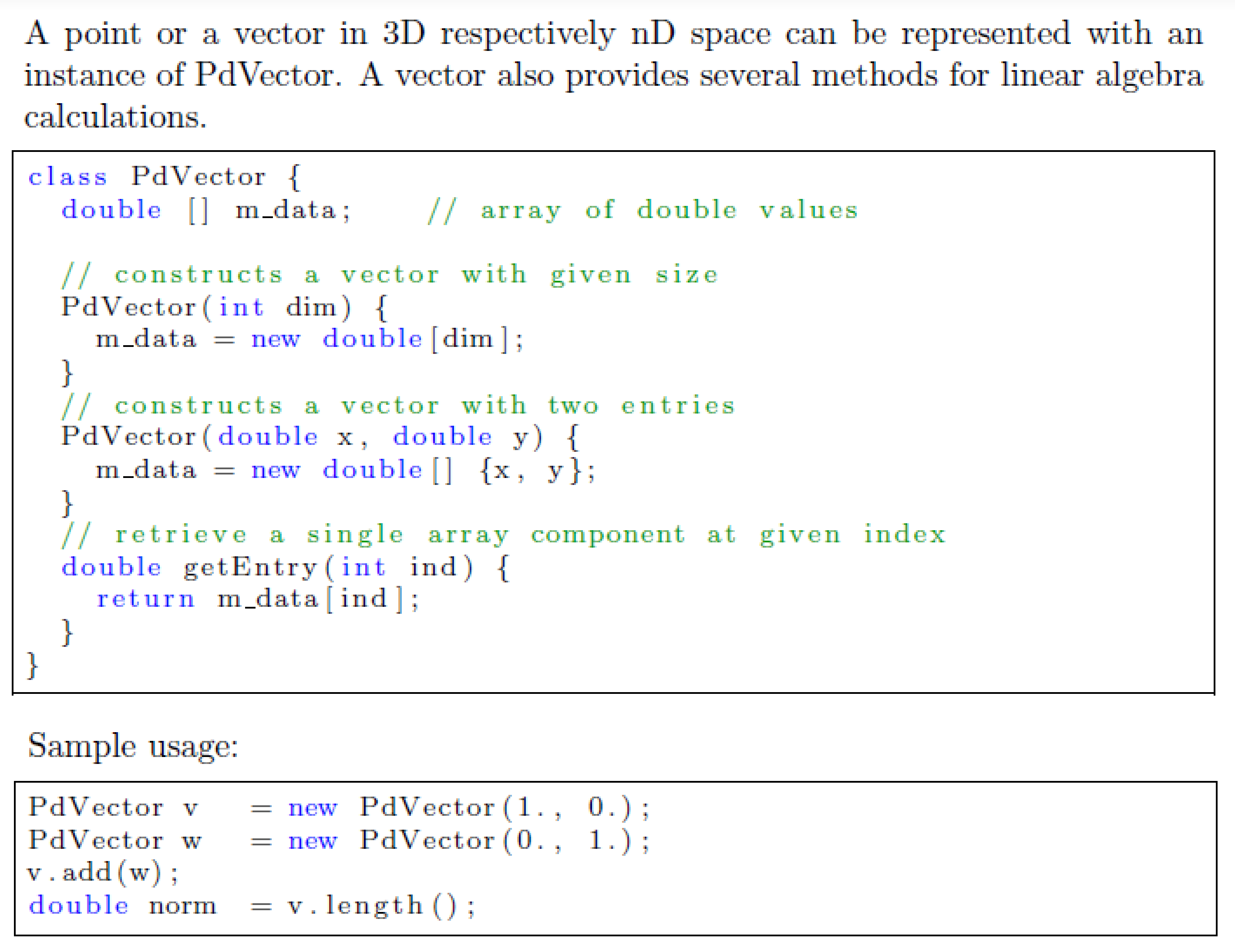
JavaView Hints

PdVector and PiVector

Vectors

- Package jv.vecmath
- jv.vecmath.PdVector, jv.vecmath.PiVector
- Maintains a double/integer array
- Double vector provides basic methods for linear algebra

PdVector



A point or a vector in 3D respectively nD space can be represented with an instance of PdVector. A vector also provides several methods for linear algebra calculations.

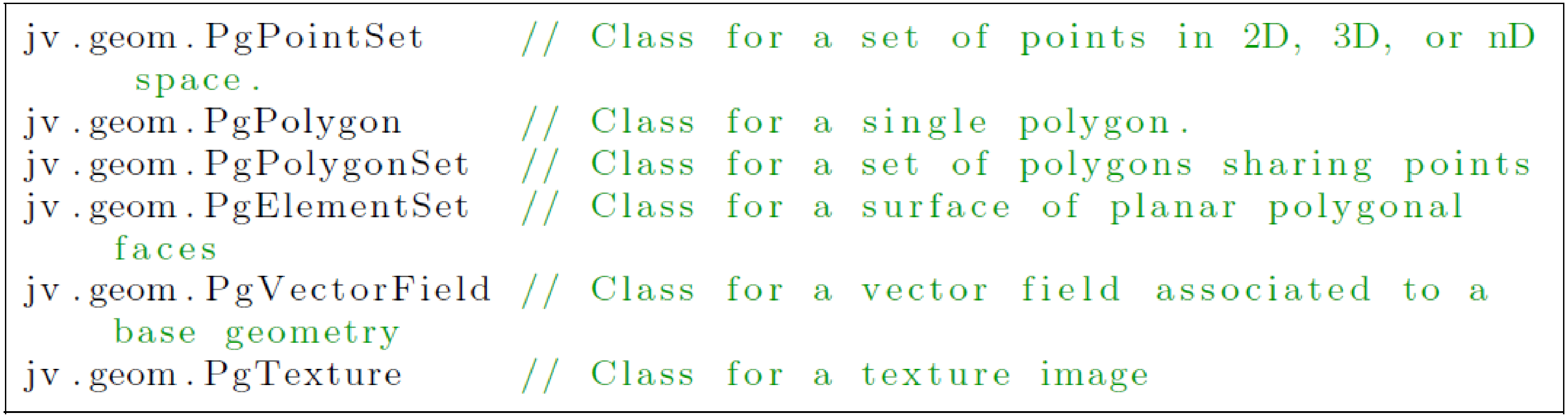
```
class PdVector {
  double [] m_data;     // array of double values

  // constructs a vector with given size
  PdVector(int dim) {
    m_data = new double[dim];
  }
  // constructs a vector with two entries
  PdVector(double x, double y) {
    m_data = new double[] {x, y};
  }
  // retrieve a single array component at given index
  double getEntry(int ind) {
    return m_data[ind];
  }
}
```

Sample usage:

```
PdVector v    = new PdVector(1., 0.);
PdVector w    = new PdVector(0., 1.);
v.add(w);
double norm   = v.length();
```

Geometries

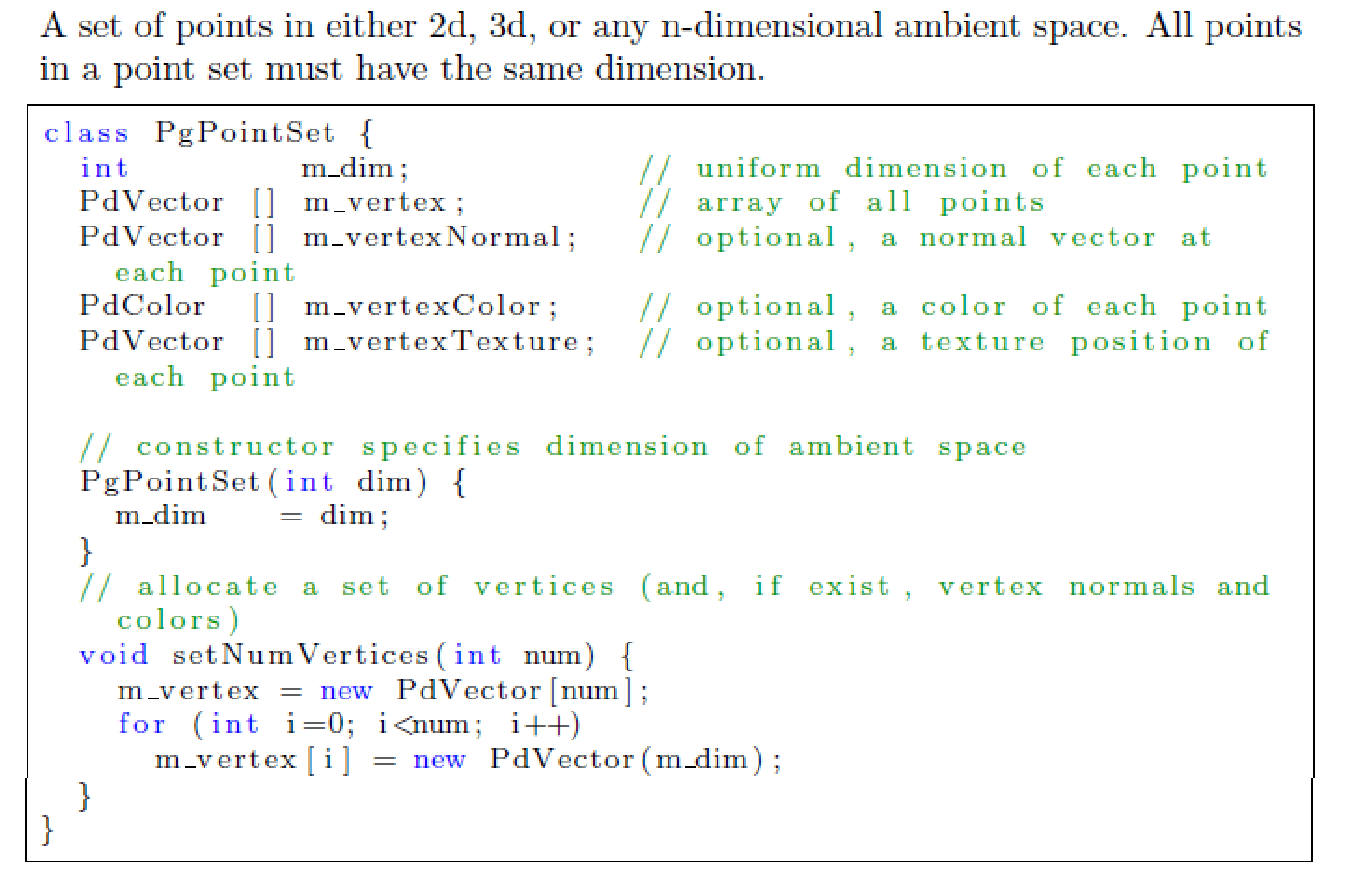
Package: jv.geom



```
jv.geom.PgPointSet     // Class for a set of points in 2D, 3D, or nD
     space.
jv.geom.PgPolygon      // Class for a single polygon.
jv.geom.PgPolygonSet   // Class for a set of polygons sharing points
jv.geom.PgElementSet   // Class for a surface of planar polygonal
    faces
jv.geom.PgVectorField  // Class for a vector field associated to a
    base geometry
jv.geom.PgTexture      // Class for a texture image
```

Point Sets

Class: jv.geom.PgPointSet

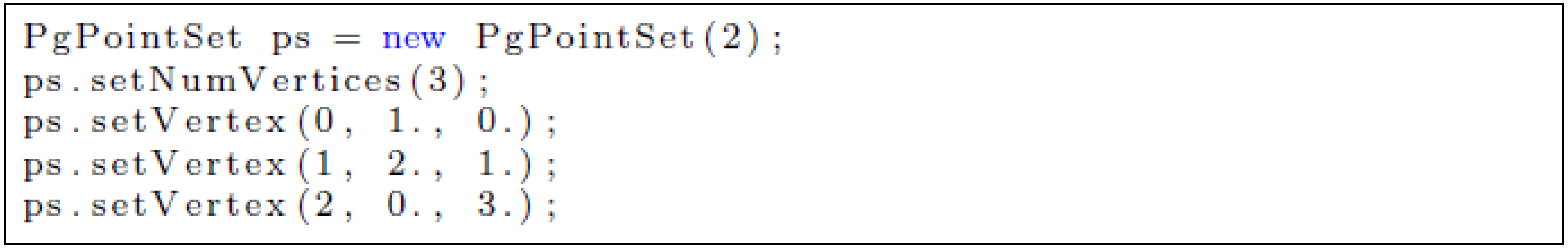


A set of points in either 2d, 3d, or any n-dimensional ambient space. All points in a point set must have the same dimension.

```
class PgPointSet {
  int            m_dim;                // uniform dimension of each point
  PdVector []  m_vertex;              // array of all points
  PdVector []  m_vertexNormal;        // optional, a normal vector at
    each point
  PdColor  []  m_vertexColor;         // optional, a color of each point
  PdVector []  m_vertexTexture;       // optional, a texture position of
    each point

  // constructor specifies dimension of ambient space
  PgPointSet(int dim) {
    m_dim     = dim;
  }
  // allocate a set of vertices (and, if exist, vertex normals and
    colors)
  void setNumVertices(int num) {
    m_vertex = new PdVector[num];
    for (int i=0; i<num; i++)
      m_vertex[i] = new PdVector(m_dim);
  }
}
```

Example

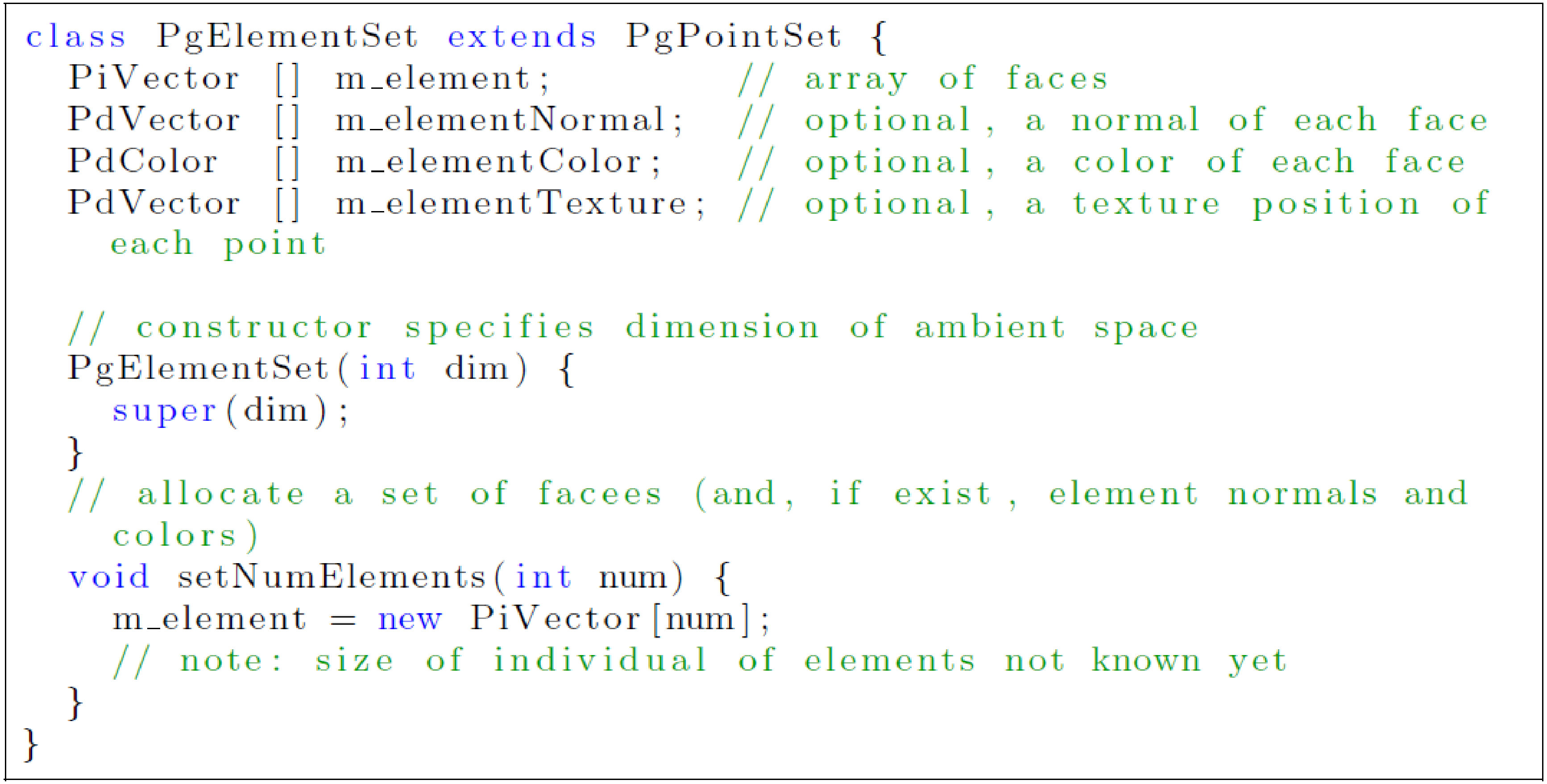
Construct a point set



```
PgPointSet ps = new PgPointSet(2);
ps.setNumVertices(3);
ps.setVertex(0, 1., 0.);
ps.setVertex(1, 2., 1.);
ps.setVertex(2, 0., 3.);
```

Meshes

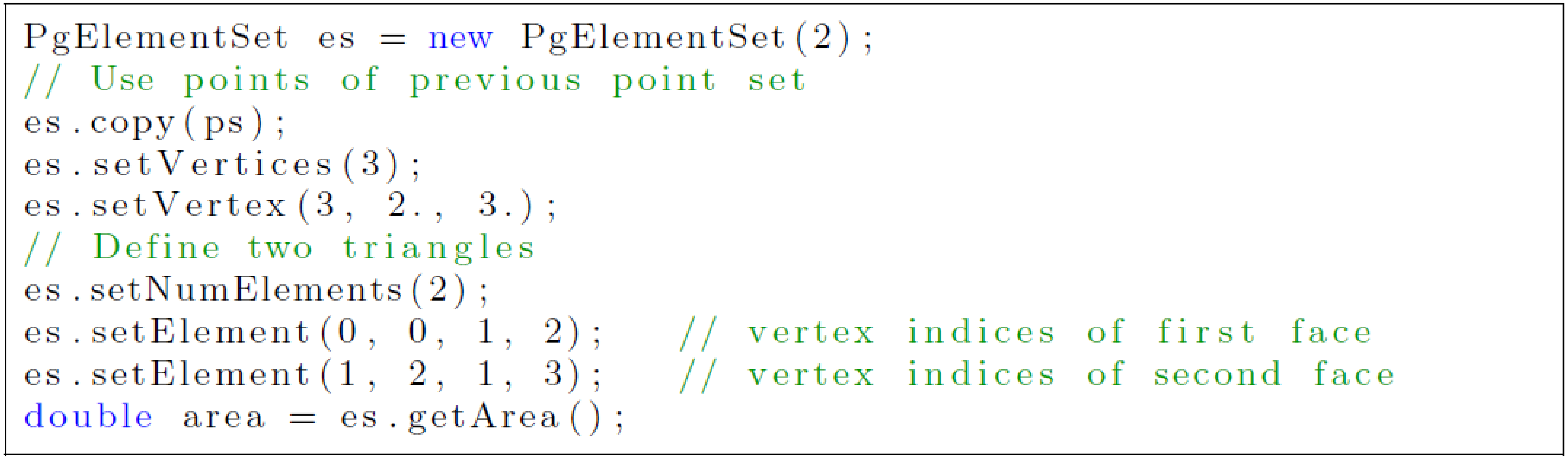
Class: jv.geom.PgElementSet



```
class PgElementSet extends PgPointSet {
  PiVector [] m_element;         // array of faces
  PdVector [] m_elementNormal;   // optional, a normal of each face
  PdColor  [] m_elementColor;    // optional, a color of each face
  PdVector [] m_elementTexture;  // optional, a texture position of
    each point

  // constructor specifies dimension of ambient space
  PgElementSet(int dim) {
    super(dim);
  }
  // allocate a set of facees (and, if exist, element normals and
    colors)
  void setNumElements(int num) {
    m_element = new PiVector[num];
    // note: size of individual of elements not known yet
  }
}
```

Example

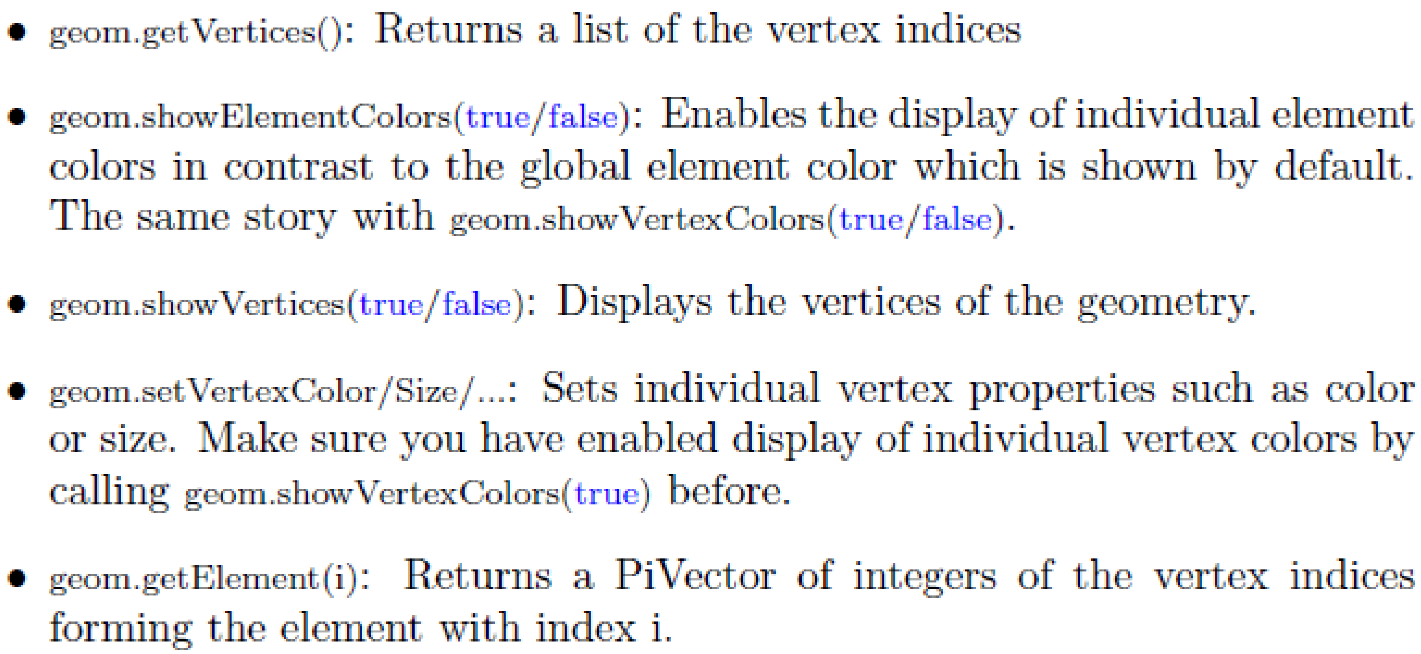
Construct a mesh



```
PgElementSet es = new PgElementSet(2);
// Use points of previous point set
es.copy(ps);
es.setVertices(3);
es.setVertex(3, 2., 3.);
// Define two triangles
es.setNumElements(2);
es.setElement(0, 0, 1, 2);    // vertex indices of first face
es.setElement(1, 2, 1, 3);    // vertex indices of second face
double area = es.getArea();
```

Example

Some examples of methods

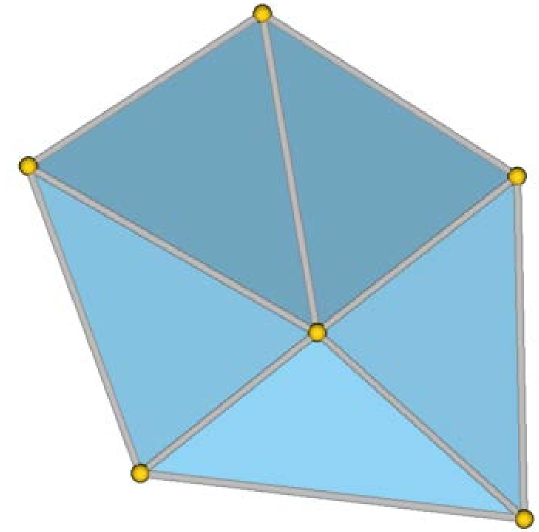


- geom.getVertices(): Returns a list of the vertex indices
- geom.showElementColors(true/false): Enables the display of individual element colors in contrast to the global element color which is shown by default. The same story with geom.showVertexColors(true/false).
- geom.showVertices(true/false): Displays the vertices of the geometry.
- geom.setVertexColor/Size/...: Sets individual vertex properties such as color or size. Make sure you have enabled display of individual vertex colors by calling geom.showVertexColors(true) before.
- geom.getElement(i): Returns a PiVector of integers of the vertex indices forming the element with index i.

Vertex Star

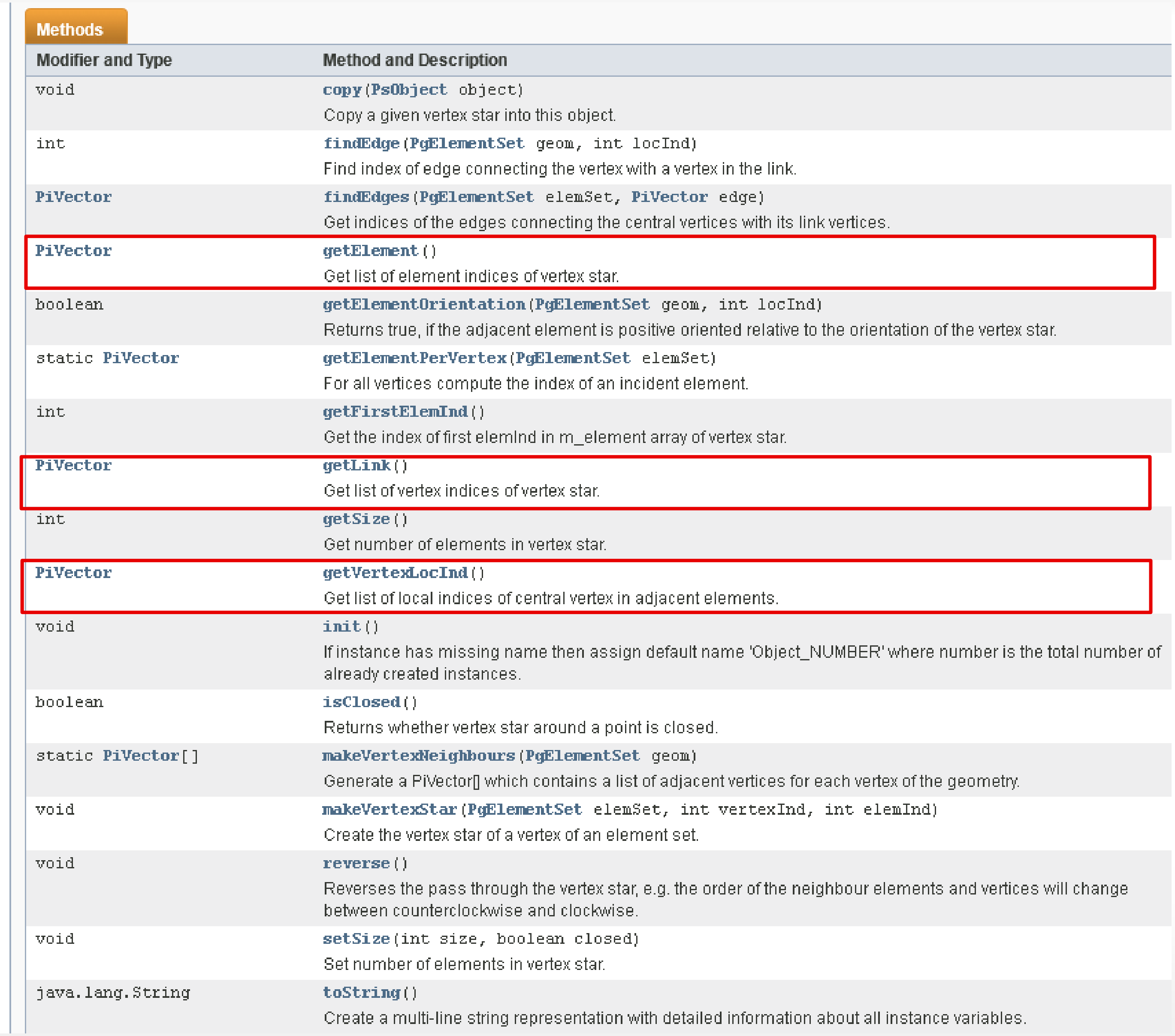
Class: jvx.geom.PgVertexStar

- Provides 1-ring neighbor information for a vertex
  - Adjecent vertices (method: getLink())
  - Adjecent elements (getElements())
  - Index of center vertex in adjecent elements (getVertexLocInd())
- Class util.Util provides a static method that returns the vertex stars of all vertices of a mesh



Star of a vertex

Vertex Star



Methods

Modifier and Type	Method and Description
void	copy(PsObject object) Copy a given vertex star into this object.
int	findEdge(PgElementSet geom, int locInd) Find index of edge connecting the vertex with a vertex in the link.
PiVector	findEdges(PgElementSet elemSet, PiVector edge) Get indices of the edges connecting the central vertices with its link vertices.
PiVector	getElement() Get list of element indices of vertex star.
boolean	getElementOrientation(PgElementSet geom, int locInd) Returns true, if the adjacent element is positive oriented relative to the orientation of the vertex star.
static PiVector	getElementPerVertex(PgElementSet elemSet) For all vertices compute the index of an incident element.
int	getFirstElemInd() Get the index of first elemInd in m_element array of vertex star.
PiVector	getLink() Get list of vertex indices of vertex star.
int	getSize() Get number of elements in vertex star.
PiVector	getVertexLocInd() Get list of local indices of central vertex in adjacent elements.
void	init() If instance has missing name then assign default name 'Object_NUMBER' where number is the total number of already created instances.
boolean	isClosed() Returns whether vertex star around a point is closed.
static PiVector[]	makeVertexNeighbours(PgElementSet geom) Generate a PiVector[] which contains a list of adjacent vertices for each vertex of the geometry.
void	makeVertexStar(PgElementSet elemSet, int vertexInd, int elemInd) Create the vertex star of a vertex of an element set.
void	reverse() Reverses the pass through the vertex star, e.g. the order of the neighbour elements and vertices will change between counterclockwise and clockwise.
void	setSize(int size, boolean closed) Set number of elements in vertex star.
java.lang.String	toString() Create a multi-line string representation with detailed information about all instance variables.

Vertex Star

Computing smoothing methods

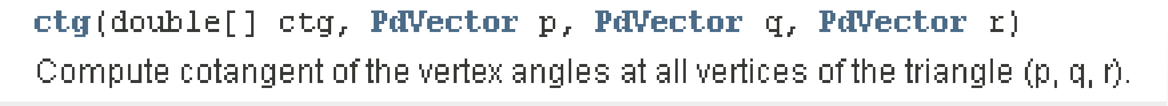
- Vertex star can be helpful for computing the average of the neighbor vertices and the mean curvature vector

Mean curvature vector

$$\vec{H}_h(x_i) = \frac{3}{2\mathrm{area}\big(\mathrm{star}(x_i)\big)} \sum_{x_j \in link(x_i)} \big(\cot\alpha_{ij} + \cot\beta_{ij}\big)\,(x_i - x_j)$$

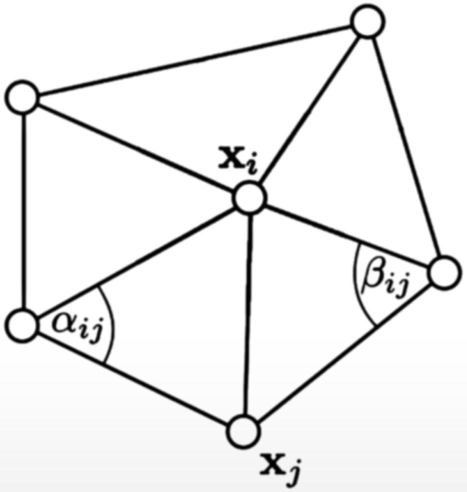
Hint: jv.vecmath.PuVectorGeom

- Provides methods for computing cotangents



```
ctg(double[] ctg, PdVector p, PdVector q, PdVector r)
```

Compute cotangent of the vertex angles at all vertices of the triangle (p, q, r).



Matrices

Dense matrices

- Class: jv.vecmath.PdMatrix

Sparse matrices

- Class: jvx.numeric.PnSparseMatrix
- Solver for sparse linear systems
  - Conjugate Gradients (java): jvx.numeric.PnConjugateGradientMatrix
  - Sparse Direct Solver: dev6.numeric.PnMumpsSolver (external library via java native interface, only Windows 64bit dlls)

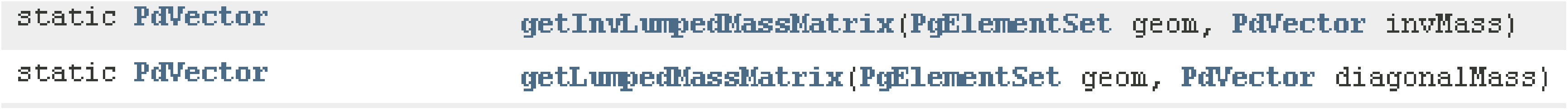
Matrices M and S

- Class: jvx.numeric.PnMassMatrix
- Class: jvx.numeric.PnStiffDiriConforming

Mass Matrix

Mass matrix

- Class: jvx.numeric.PnMassMatrix
- To get the diagonal matrix use methed: useLumpedMass()
- Diagonal matrix (and inverse) can be generated as a PdVector that contains the diagonal entries



static PdVector	getInvLumpedMassMatrix(PgElementSet geom, PdVector invMass)
static PdVector	getLumpedMassMatrix(PgElementSet geom, PdVector diagonalMass)

Example:

```
PnMassMatrix mass = new PnMassMatrix(geom, true);
```

PgElementSet

Get diagonal matrix

Stiffness Matrix

Stiffness Matrix (*S* in the lecture)

- Class: jvx.numeric.PnStiffDiriConforming

Example:

PnStiffDiriConforming stiff = new PnStiffDiriConforming(geom);

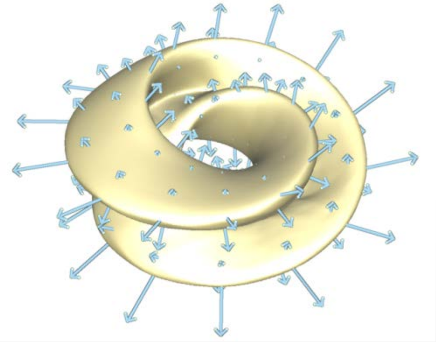
Mean Curvature Vectors

- The mean curvature vectors can be computed using M and S

  Discrete mean curvature vector is $\vec{H}_h \in {S_h}^3$

  $$\vec{H}_h = M^{-1} S x$$

  - Matrix must be applied to the x,y,and z coordintates individually



Practical Hints

Workshop

- The folder workshop includes an example workshop that you can modify to design your own workshop

Menu

- The folder menu contains a file that you can edit to integrate your workshop into the program’s menu

Meshes

- Some meshes for test your implementation are contained in the model.zip file
- To add noise to a mesh, select Method->Effect->Noise from the menu.