3D Visualisation

The given implementation displays tetrahedrons, triangles, edges and vertices on a jMonkey window. A complementary JavaFX window offers the possibility to select and deselect objects that get displayed on the jMonkey window.

Data Input

The class App contains the main-method. The class AppFX is embedded into App and calls javafx. The class AppJME is embedded into App and calls a jMonkey Window. (see Figure 1)
The data input for this implementation is given Within the method createMeshExample in the App. For the input an object of the type Mesh has to be created. Afterwards tetrahedrons have to be added. This can happen in two ways. A Tetrahedron3D object can be added to the Mesh object and assigned to a cell via the method addTet(Tetrahedron3D, int). The other option is to solely add a Tetrahedron3D object to the mesh. This object than has the option to be added to a cell via the method addExistingTetToCell (Tetrahedron3D, int). Cells do not have to be initialised before tetrahedrons are added to them, they automatically get initialised.
With every tetrahedron its four triangles, six edges and four vertices are automatically added to the mesh. If wanted boundary elements can be marked as such. The class Mesh contains the method markBoundary for this purpose. It checks boundary criteria for the specific example mesh. Accordingly, it has to be adapted to the input given in createMeshExample.

The simpleInitApp method of the static class AppJME contains the colours used to display tetrahedrons, triangles, edges and vertices. These can be adapted to ones likings. The setScene method of the class Presenter contains an enumeration of the type InitialView as parameter among other things. The selection of different values for this enumeration determines which of the boundary elements get displayed when launching the program.

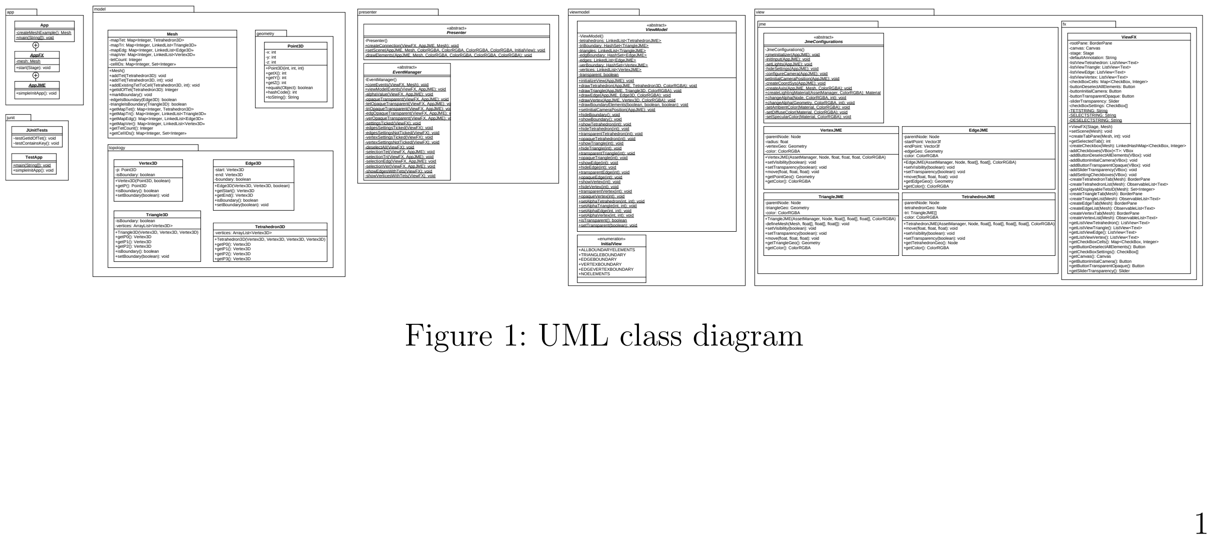

Figure 1: UML class diagram

Usage

In the jMonkey view the camera can be moved through keys:
- W: move forward
- S: move backward
- A: move left
- D: move right
- Q: move up (in positive z-direction)
- Z or Y: move down (in negative z-direction).

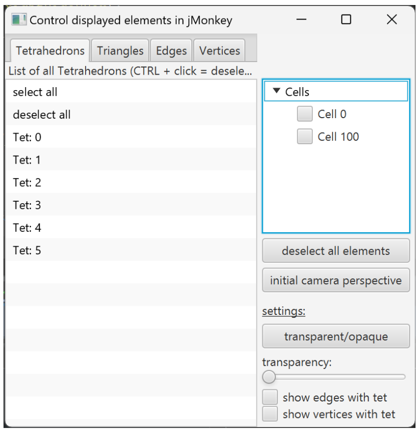

Figure 2: JavaFX view

The JavaFX view shown in Figure 2 allows the user to navigate to different tabs to choose between possible objects to select. In every Tab the option to select all objects and deselect all objects is given. Objects can be selected solely, but also in bunches. The shift key and CTRL key in combination with clicks select in the standard manner known from other programs:
- shift + click: select all elements between the last selection and this selection
- CTRL + click on unselected element: add this selection to the already selected elements
- CTRL + click on selected element: remove this selection from the already selected elements.

If cells are specified in the input they get displayed on the right side of the JavaFX view. Cells limit the selectable elements in the specified tabs. Multiple cells can be selected. If no cell is selected all elements are displayed in the tabs.
The right side of the view also offers a button to deselect all elements. Additionally, there is a button to get redirected to the initial camera perspective and a button to switch the material between being opaque and transparent. A slider beneath the buttons controls the level of transparency if the material is in transparent mode. Furthermore, there is the option to show edges and/or vertices with tetrahedrons. If these checkboxes are ticked with every displayed tetrahedron its edges and/or vertices are also displayed on the jMonkey view.

Resources

With the implementation the uml class diagram created in UMLet is provided. If this visualisation is ought to be extended that file can be adapted to the new implementation.