ThreeSpace
IMPLEMENTATION DOCUMENT

Adam Chodorowski & John Nilsson

June 28, 2001

Contents

1 Introduction **3**

1.1 Purpose . . . 3

1.2 Definitions . . . 3

2 Package overview **4**

2.1 threespace.scene . . . 4

2.1.1 Scene . . . 4

2.1.2 SceneListener . . . 4

2.1.3 Camera . . . 4

2.1.4 MovableObject . . . 4

2.1.5 ColorableObject . . . 5

2.1.6 SceneObject . . . 5

2.1.7 Shape . . . 5

2.1.8 Geometry . . . 5

2.1.9 KeyBehaviors . . . 5

2.1.10 MouseBehaviors . . . 5

2.2 threespace.scene.event . . . 5

2.2.1 ColorChangeEvent . . . 5

2.2.2 ColorChangeListener . . . 6

2.2.3 PositionChangeEvent . . . 6

2.2.4 PositionChangeListener . . . 6

2.2.5 RotationChangeEvent . . . 6

2.2.6 RotationChangeListener . . . 6

2.3 threespace.gui . . . 6

2.3.1 AboutDialog . . . 6

2.3.2 SplashDialog . . . 6

2.3.3 CreateDialog . . . 7

2.3.4 FilterDialog . . . 7

2.3.5 MenuPanel . . . 7

2.3.6 MenuActionEvent . . . 7

2.3.7 MenuActionListener . . . 7

2.3.8 ToolBarPanel . . . 7

2.3.9 PropertyPanel . . . 7

2.3.10 ViewPanel . . . 8

2.3.11 ColorSelector . . . 8

2.3.12 SliderGroup . . . 8

2.3.13 TextFieldGroup . . . 8

2.3.14 ITextField . . . 8

2.3.15 DoubleField . . . 8

2.3.16 IntegerField . . . 8

2.3.17 ValidatingDocument . . . 8

2.3.18 PluginListModel . . . 8

2.3.19 ErrorMessage . . . 8
2.4 threespace.gui.editor . . . 9
2.4.1 ColorEditor . . . 9
2.4.2 NameEditor . . . 9
2.4.3 PositionEditor . . . 9
2.4.4 RotationEditor . . . 9
2.5 threespace.plugin . . . 9
2.5.1 Circle2D . . . 9
2.5.2 Function3D . . . 9
2.5.3 HeightColorFilter . . . 9
2.5.4 Hemisphere3D . . . 9
2.5.5 Mandelbrot3D . . . 9
2.5.6 Plane2D . . . 10
2.5.7 SignColorFilter . . . 10
2.5.8 Sphere3D . . . 10
2.5.9 Terrain3D . . . 10
2.6 threespace.plugin.type . . . 10
2.6.1 Plugin . . . 10
2.6.2 ShapeCreator . . . 10
2.6.3 Filter . . . 10
2.6.4 ColorFilter . . . 10
2.6.5 PositionFilter . . . 10
2.6.6 RotationFilter . . . 10
2.6.7 StructureFilter . . . 11
2.6.8 ObjectLoader . . . 11
2.6.9 ObjectSaver . . . 11
2.6.10 SceneLoader . . . 11
2.6.11 SceneSaver . . . 11
2.6.12 ViewExporter . . . 11
2.7 threespace.plugin.manager . . . 11
2.7.1 ListChangeEvent . . . 11
2.7.2 ListChangeListener . . . 11
2.7.3 PluginList . . . 11
2.7.4 PluginLoader . . . 12
2.7.5 PluginManager . . . 12
2.8 threespace.jel . . . 12
2.8.1 Calculator . . . 12
2.8.2 Variables . . . 12
2.9 threespace.util . . . 12
2.10 DEBUG . . . 12

3 Diagrams **13**

4 Javadoc **14**

Chapter 1

Introduction

1.1 Purpose

This document aims to describe the current implementation of ThreeSpace; what packages and classes it consists of and how they interact.

1.2 Definitions

Please see the requirements document and design document for a full list of definitions and abbreviations used in this document.

Chapter 2

Package overview

ThreeSpace consists of 9 different packages and the top-level class called Application. Application handles the startup of the program, making sure to instantiate the different classes that are required for correct operation. It also handles the top-level GUI (ie. the main window), even though most of the work is delegated to classes in threespace.gui.

2.1 threespace.scene

This package contains classes for handling the composition of the scene, the visual objects it contain (and their composition) and visualization using Java3D.

The following classes are available:

2.1.1 Scene

The Scene class handles the Scene with all its SceneObject, the Camera and creates the Component which can be added to the ViewPanel.

2.1.2 SceneListener

An interface for a listener that listens for Scene events.

2.1.3 Camera

The Camera class wraps around the Java3D View Model. Implmenting MovableObject allows it to be modified as a regular object with rotation and position. A Camera object can be considered as a camera looking at a center point from a given angle from a given distance. It can thus be defined using:

1. A center point which the camera is looking at
2. A distance from the center point
3. Three angles which define the orientation and direction from which the camera is looking

The distance from the center and the position at which the center appears on the canvas are handled by the MovableObject methods. The three angles are also handled by MovableObject. Choosing the point to center around is left as a method which the GUI can access if it so chooses.

2.1.4 MovableObject

A MovableObject can move and rotate.

2.1.5 ColorableObject

An abstract class which extends MovableObject and adds color methods.

2.1.6 SceneObject

A SceneObject holds the two shapes which contain geometry-information. One shape is before filters have applied and the others is after they have been applied. The SceneObject is added to the Scene's list of objects.

2.1.7 Shape

The Shape class encapsulates the Shape's geometry and the properties that control how the Shape is viewied. It also controls the rotation and positioning of the object.

2.1.8 Geometry

The Geometry class is the wrapper around the Java3D Geometry data. The Java3D data is in the form of IndexedTriangleArray and thus all constructors and access is on the form of this structure. The arrays that are included in the structure are:

1. Vertices - all vertices in the 3D object on the form of {x1,y1,z1,x2,y2,z2,x3,y3,z3}
2. Colors - the color of the vertices specified in the Vertices array. {red1,green1,blue1,r2,g2,b2}
3. Triangles - Defines the connection between vertices {triangle1vertex1,triangle1vertex2,triangle1vertex3, triangle2vertex1,triangle2vertex2,triangle2vertex3}
4. Normals - automatically created. Creates a normal for each triangle.

2.1.9 KeyBehaviors

This class handles the KeyboardEvents that the Canvas receives. It handles movement of the Centerpoint for the camera.

2.1.10 MouseBehaviors

This class handles the MouseEvents that the Canvas receives. It handles movement and rotation of the Camera and selected SceneObject. It also handles selection of a objects and selection of CenterAround-object.

2.2 threespace.scene.event

This package contains classes and interfaces for implementing notifications when things (rotation, position, color, etc.) are changed in the Scene or in a SceneObject. This makes it possible for non-related classes (eg. that draw the GUI handling) to update their information without direct intervention of the scene classes.

The following classes are available:

2.2.1 ColorChangeEvent

This class is used to notify interested parties that the color of the event source (ie. a ColorableObject) has changed.

2.2.2 ColorChangeListener

This is an interface for classes to implement if they wish to be notified about ColorChangeEvents from some kind of event source.

2.2.3 PositionChangeEvent

This class is used to notify interested parties that the position of the event source (ie. a MovableObject) has changed.

2.2.4 PositionChangeListener

This is an interface for classes to implement if they wish to be notified about PositionChangeEvents from some kind of event source.

2.2.5 RotationChangeEvent

This class is used to notify interested parties that the rotation of the event source (ie. a MovableObject) has changed.

2.2.6 RotationChangeListener

This is an interface for classes to implement if they wish to be notified about RotationChangeEvents from some kind of event source.

2.3 threespace.gui

This package contains the main bulk of classes and interfaces that are used for creating the GUI. This ranges from flexible GUI components that are meant for reuse in other classes and specific classes that handle a single domain (eg. CreateDialog that handles the object- creation dialog).

The following classes are available:

2.3.1 AboutDialog

This class implements a simple about dialog that displays the following information about the application:

1. Logotype image.
2. Version information.
3. List of authors.

It also provides a button for closing the window. See the design document for a scetch of how the dialog looks (or compile and run the code).

2.3.2 SplashDialog

This class implements a simple splash dialog, useful for providing feedback when the application is started. It look really nice, too. The dialog provides the following:

1. Logotype image.
2. Progress bar.

3. Description of current event.

See the design document for a scetch of how it looks (or compile and run the code). All methods provided by this class affect only the display of the GUI, nothing else.

2.3.3 CreateDialog

The Dialog which ShapeCreator plugins are shown in. The dialog includes the ShapeCreator's option-panel.

2.3.4 FilterDialog

The dialog used to control a specific filter list of a scene object. It allows new Filters to set their properties. Also includes functions to remove and add filters to the filter list.

2.3.5 MenuPanel

The MenuPanel populates the menus with the plugins and the standard menuitems.

2.3.6 MenuActionEvent

An extension to ActionEvent which stores an extra Object parameter which can then be retrieved through the getParameter().

2.3.7 MenuActionListener

An interface for a listener capable of receiving the special MenuActionEvents.

2.3.8 ToolBarPanel

This class implements a specially customised version of JToolbar for embedding in the main window of ThreeSpace. It provides buttons for the following actions:

1. New scene
2. Load scene
3. Save scene
4. Save scene as
5. Switch to Camera State
6. Switch to Object State
7. Switch to Vertex State
8. Switch to rendering mode 1
9. Switch to rendering mode 2
10. Switch to rendering mode 3
11. Switch to rendering mode 4

2.3.9 PropertyPanel

The PropertyPanel handles the CameraPanel and the ObjectPanel. When the scene is in Camera-mode the CameraPanel is displayed and when the scene is in Object-mode the ObjectPanel is displayed.

The CameraPanel includes controls for changing the camera position and viewing direction and also has a control for adding bookmarks.

The ObjectPanel includes controls for changing the objects position, rotation, name, color and the filters which it utilizes.

2.3.10 ViewPanel

The ViewPanel is a panel which contains the Canvas3D component (ie. that renders the view).

2.3.11 ColorSelector

The GUI component which lets the user select a specific color through either RGB or HSV values. The getColor method returns the RGB-values.

2.3.12 SliderGroup

A Panel which includes a number of sliders that go between specified min and max values.

2.3.13 TextFieldGroup

A Panel which handles a number of textfiels and puts a plus and minus button next to them.

2.3.14 ITextField

An extension to JTextField which sends an ActionEvent when the text field loses focus. The regular JTextField does not do this, which can result in that values aren't properly updated if you're not careful.

2.3.15 DoubleField

An extension to JTextField which only allows doubles to be written in the text field.

2.3.16 IntegerField

An extension to JTextField which only allows integers to be written to the text field.

2.3.17 ValidatingDocument

An abstract class for validating text before the text is allowed to enter a TextField.

2.3.18 PluginListModel

A List model which handles the PluginLists we have for Color, Structure, etc. It updates its values whenever the PluginList changes, automatically.

2.3.19 ErrorMessage

A class used by filters and plugins to show error messages

2.4 threespace.gui.editor

This package contains flexible classes that are used for editing a scene object's different attributes. These classes are "embed-and-forget", ie. you only have to give them a reference to a scene object and embed them in a GUI, and they will handle all interaction between the user and the scene object automatically by themselves.

The following classes are available:

2.4.1 ColorEditor

The ColorEditor panel is used for changing the color of an object. It uses a ColorSelector for selecting the color.

2.4.2 NameEditor

The NameEditor panel is used for changing the name of an object. It uses a ITextField.

2.4.3 PositionEditor

The editor used for changing the position of the camera or an object.

2.4.4 RotationEditor

The editor used for changing the rotation of the camera or an object.

2.5 threespace.plugin

This package contains all concrete plugins that have been implemented. These can be plugins for creating new scene objects, filtering plugins, savers, loaders, etc. All must implement some interface as defined in threespace.plugin.type.*.

The following classes are available:

2.5.1 Circle2D

This class implements a ShapeCreator plugin that generates a plain circle with a specified radius.

2.5.2 Function3D

This class implements a ShapeCreator plugin that generates a 3D shape that visualizes a mathematical function, eg. $z = f(x,y)$.

2.5.3 HeightColorFilter

This class implements a ColorFilter which sets the color according to the height of the object.

2.5.4 Hemisphere3D

This class implements a ShapeCreator plugin that generates a hemispere.

2.5.5 Mandelbrot3D

This class implements a ShapeCreator plugin that generates a mandelbrot image in 3D given some parameters.

2.5.6 Plane2D

This class implements a ShapeCreator plugin that generates a 2D plane.

2.5.7 SignColorFilter

This plugin implements a ColorFilter that gives all vertices with positive z-coordinates a specific color.

2.5.8 Sphere3D

This class implements a ShapeCreator plugin that generates a 3D spere with a specific radius.

2.5.9 Terrain3D

This class implements a ShapeCreator plugin that generates a randomly created shape in the form of a mountain.

2.6 threespace.plugin.type

This package contains interfaces for all kinds of plugins supported by ThreeSpace. All that a specific plugin must do is implement one if these interfaces, and it will automatically be instantiated and available for use in ThreeSpace.

The following classes are available:

2.6.1 Plugin

This is the minimal common interface that all plugins for ThreeSpace must implement. Ofcourse plugins should implement subinterfaces of this interface, since it doesn't provide any actual functionality.

2.6.2 ShapeCreator

This is an interface for plugins that create new shapes from scratch.

2.6.3 Filter

This is the top-level interface for any filter plugins that modify SceneObjects in some way. Note that concrete plugins must implement a subclass of this interface; simply implementing this interface is not very useful.

2.6.4 ColorFilter

This is an interface for filter plugins that modify a SceneObject's color.

2.6.5 PositionFilter

This is an interface for filter plugins that modify a SceneObject's position.

2.6.6 RotationFilter

This is an interface for filter plugins that modify a SceneObject's rotation.

2.6.7 StructureFilter

This is an interface for filter plugins that modify a SceneObject's structure, ie. the vertices, triangles and/or polygons.

2.6.8 ObjectLoader

This is an interface for plugins that can load a SceneObject from disk.

2.6.9 ObjectSaver

This is an interface for plugins that can save an SceneObject to disk, possibly for later retrieval.

2.6.10 SceneLoader

This is an interface for plugins that can load a Scene from disk.

2.6.11 SceneSaver

This is an interface for plugins that can save a Scene out to disk, possibly for later retrieval.

2.6.12 ViewExporter

This is an interface for plugins that export the currently rendered view to a file. This generally means encoding the 2D image as a JPEG, GIF, PNG or a similar format; although this is up to the plugin to decide.

2.7 threespace.plugin.manager

This package contains classes and interfaces for handling and loading the plugins. This consists of a custom class-loader for loading and instantiating classes from disk, a manager for handling the collection of plugins (scanning the plugin-directory, grouping plugins by type, etc) and various supporting classes.

The following classes are available:

2.7.1 ListChangeEvent

This class is used to notify interested parties that the state of the event source has changed, and how. It is intended for use with different kinds of indexed lists, and you can specify if a specific list element has been added or removed.

2.7.2 ListChangeListener

This is an interface for classes to implement if they wish to be notified about ListChangeEvents from some kind of event source.

2.7.3 PluginList

This class implements an abstract collection of Plugins of a specific type. It is does not implement the full List interface, since that seemed unneccesary for the intended use. Apart from methods for operating on the list, it can also notify interested parties using ListChangeEvents if it is changed in some way.

2.7.4 PluginLoader

This class implements a custom class-loader for dynamic loading of plugins.

2.7.5 PluginManager

This class implements a high-level handler of collections of different types of Plugins. It automatically scans the plugin-directory and uses the PluginLoader to load any class-files it can find there. It also provides an API for getting at lists of Plugins organized after type.

2.8 threespace.jel

This package contains classes for interfacing with GNU JEL (Java Expression Library) which support for parsing and evaluating mathematical expressions. This is then used by the various plugins that support mathematical formulae.

The following classes are available:

2.8.1 Calculator

This class interfaces with GNU JEL (Java Expression Library) for parsing and evaluation mathematical formulae.

2.8.2 Variables

This class is just a wrapper around variables that can be used in the expressions evaluated by Calculator.

2.9 threespace.util

This package is meant for different utility-classes that don't fit in anyware else.

There is currently only a single class available:

2.10 DEBUG

This class provides static methods for printing debug output to the console. This way, you just have to disable the output in this class to disable all debug output in the program.

Chapter 3

Diagrams

We have created some diagrams to illustrate the different packages of the program and how the different classes interact and subclass each other. Note that some of these files are big.

A diagram that illustrates the GUI package can be found at:
http://www.d.kth.se/ d00-jni/progstil/implementation/gui.png

A diagram that illustrates the plugin package can be found at:
http://www.d.kth.se/ d00-jni/progstil/implementation/plugin.png

A diagram that illustrates the scene package can be found at:
http://www.d.kth.se/ d00-jni/progstil/implementation/scene.png

And, if you want to see it all in it's whole glory, the following diagram shows all packages and their classes together:
http://www.d.kth.se/ d00-jni/progstil/implementation/threespace.png

Chapter 4

Javadoc

Please se the Javadoc for a complete reference of the methods and classes.
http://www.d.kth.se/ d00-jni/progstil/javadoc/