16 February, 2005

Development of a pipeline designer for VTK using Javabeans

Andrei Jalba

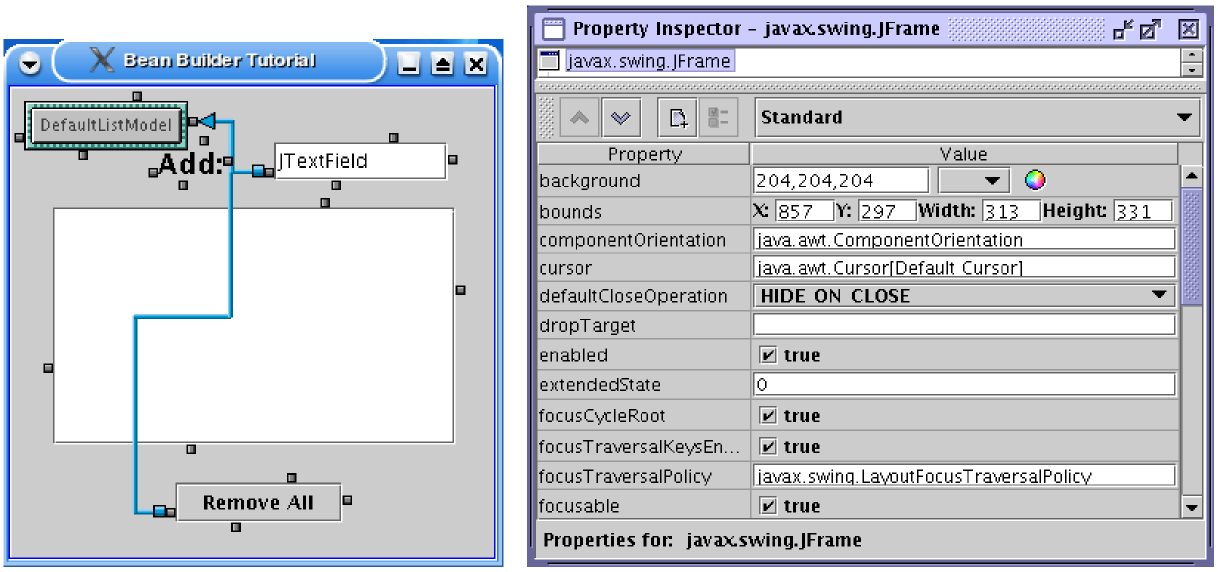
1 Description of the project

The Visualization ToolKit (VTK) is an open source, freely available software system for 3D computer graphics, image processing, and visualization used by thousands of researchers and developers around the world. VTK consists of a C++ class library, and several interpreted interface layers including Tcl/Tk, Java, and Python (see `http://public.kitware.com/VTK/`). In fact, VTK has becoming a state-of-the art visualization system, providing a wide range of functionality, encapsulated in over 1500 C++ classes. Unfortunately, unlike other commercial visualization systems (i.e. Amira, AVS), the VTK does not provide an easy to use designer for visual construction of VTK pipelines.

Aim of the project. The aim of the project is to develop a pipeline designer for VTK using *Javabeans* components, which will enable (unexperienced) users to develop graphically VTK pipelines. For this, we need (i) to turn the C++ VTK classes into Java classes, using the *Java Native Interface* (JNI), and (ii) convert the Java classes into *Javabeans* components. Note that, VTK provides a basic binding for Java (i.e. the first step has been implemented in VTK), however, the (automatically) generated classes are not Javabeans components.

Approach. The Javabeans architecture is a modern and dynamic component model designed for building independent software components from which programmers easily assemble larger applications. A Javabeans component is an object that conforms to a comunication and configuration protocol, as prescribed by the *JavaBeans specification* (available on Sun's Java site). This specification prescribes programming conventions and dynamic discovery mechanisms that (i) minimize the design and implementation effort for small components, while (ii) fully supporting the design. For example, using the Javabeans model one can implement a progress bar in 200 lines of code or less. Another advantage of this extensible architecture is that any Javabeans component can be used in any visual programming environment (or IDE, RAD, etc.). To minimize the memory overhead inherent when developing large Java projects, we plan to use a very simple yet quite functional, open source RAD system – the *Bean Builder*. The Bean Builder is a visual programming environment that demonstrates the assembly of applications by joining live instances of JavaBeans, see Fig. 1.

Although the Bean Builder is still under development, the basic functionality required by the current project is in its place. The advantages of using such an approach for the development of VTK pipelines (from the users' point of view) are: (i) easy to use graphical interface provided by the Bean Builder, (ii) no Java nor VTK low-level programming requirements, (iii) possibility to enrich the VTK pipeline with a basic GUI system, for easy interaction and changes of parameters of various VTK modules, (iv) persistence of the developed VTK pipeline, granted by the Serialization



Property Inspector - javax.swing.JFrame

javax.swing.JFrame

Standard

Property	Value
background	204,204,204
bounds	X: 857 Y: 297 Width: 313 Height: 331
componentOrientation	java.awt.ComponentOrientation
cursor	java.awt.Cursor[Default Cursor]
defaultCloseOperation	HIDE_ON_CLOSE
dropTarget	
enabled	☑ true
extendedState	0
focusCycleRoot	☑ true
focusTraversalKeysEn...	☑ true
focusTraversalPolicy	javax.swing.LayoutFocusTraversalPolicy
focusable	☑ true

Properties for: javax.swing.JFrame

Figure 1: Bean Builder a Javabeans component builder. *Left*: Event management (i.e. bean connections) window; *right*: bean properties editor. A third window is available within the Bean Builder (not shown here), which allows the user to drag-and-drop various graphical widgets (e.g. push buttons, text fields, etc.) in the design window (shown in the left image).

mechanism implicitly used by Javabeans components, (v) possibility to generate stand-alone Java applications, which encapsulate the VTK pipeline (not yet supported by the Bean Builder).

We plan to improve and extend the existing VTK binding for Java, in order to turn the Java classes into reusable Javabens components. For this, we need to (i) build a parser (similar to that included in the VTK distribution), which generates in a highly automated fashion the Javabeans wrappers around the existing VTK C++ classes, (ii) include (a subset of) the generated beans in the Bean Builder, (iii) (eventually) slightly modify the Bean Builder (its sources are available) to better accommodate the VTK Javabeans components.

2 Work programme

The work programme can be subdivided into the following work packages:

1. Exploration of the literature on: (i) Javabeans technologies, (ii) VTK architecture, (iii) parser construction using the *lex* (lexical analyzer) and *yacc* (grammar generator) tools.

2. Implementation of the parser.

3. Manual adjustment of the generated VTK Javabeans components.

4. Integration of the generated Javabeans components within the Bean Builder.

5. Development of a tutorial and several examples explaining the use of the framework.

6. Analysis and evaluation of the developed framework.

7. Writing a report which includes (i) a description of the architecture of the developed framework, (ii) a (critical) analysis and evaluation, (iii) the tutorial and examples, written previously.

3 Duration of the project

4 Requirements

Intermediate-level knowledge of C/C++, Java, and lex/yacc.