STK X Tutorial – STK Java API

Contents

Introduction

In this exercise you will gain hands-on experience using STK X to embed STK functionality in a container application created using the STK Java API.

Tutorial Source code

Eclipse projects and java source code for an incomplete and completed application can be found in the STK / STK Engine install at the following parent directory of:

<STK Install>/CodeSamples/CustomApplications/Java

Under this parent directory, the incomplete project sample that contains "TODO" comments is located in a folder called:

AWT_STK_X_Tutorial_Begin

Your goal is to replace the "TODO" comments and thus fix compilation errors in the AWT_STK_X_Tutorial_Begin project sample. You may compare your implementation against the completed project sample during or after you have finished the AWT_STK_X_Tutorial_Begin project sample. The completed project sample is located in a subdirectory of the above listed parent directory called:

AWT_STK_X_Tutorial_End

These samples should have been imported into your Eclipse workspace with the other STK Java API code samples when you followed the directions in the STK Java API documentation provided within the Integration Developer Kit Windows help file (<install>/help/integration.chm). Look for directions on loading "All" samples. Note you must load all Eclipse samples under <install>/CodeSamples in order to have the JRE configured properly. Remember if you do not have administrator privileges, it is recommended to copy the entire <install>/CodeSamples directory to an area where you have user write privileges in order to edit the sample code.

Add the Java UI plumbing

1) In your Eclipse workspace that has imported the STK Java API code samples, locate the CustomApp_AWT_STK_X_Tutorial_Begin sample and open it.

2) Locate the Main.java file and double click it to start the Java file text editor in Eclipse.

3) Locate the "TODO: #1" comment. Underneath this comment, make the main class extend from a JFrame. An example would be:

```
extends JFrame
```

4) Locate the "TODO: #2" comment and place the MainRunnable on the AWT Event Queue for processing in a non-blocking invocation using javax.swing.SwingUtilities class. An example would be:

```
SwingUtilities.invokeLater(r);
```

5) Locate the "TODO: #3" comment. Underneath this comment, set the default close operation of the JFrame to DISPOSE_ON_CLOSE and add a window listener. An example would be:

```
this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
this.addWindowListener(new MainWindowAdapter());
```

Add the STK Java API functionality to your application

1) At the top of the Main.java file above the class definition, locate the "TODO: #4". Underneath this comment, add some STK Java API imports for AGI logging, AWT core, AWT STK X controls, and STK Engine Custom Application initialization and uninitialization. An example would be:

```
import agi.logging.*;
import agi.core.*;
import agi.core.awt.*;
import agi.stkx.*;
import agi.stkx.awt.*;
import agi.stkengine.*;
```

2) Find the "TODO: #5" comment, it is under the "Main" class definition, next to the data member list. Underneath this comment, declare a private STK X Application Class data member variable called m_AgSTKXApplicationClass. An example would be:

```
private AgSTKXApplicationClass   m_AgSTKXApplicationClass;
```

3) Find the "TODO: #6" comment. Underneath this comment, declare a private Globe Control Class data member variable called m_AgGlobeCntrlClass. An example would be:

```
private AgGlobeCntrlClass m_AgGlobeCntrlClass;
```

4) Find the "TODO: #7" comment. Underneath this comment, declare a private Map Control Class data member variable call m_AgMapCntrlClass. An example would be:

```
private AgMapCntrlClass   m_AgMapCntrlClass;
```

5) Find the "TODO: #8" comment. Underneath this comment, initialize the STK Java API's Java Native Interface (JNI) connection for use on the AWT/Swing Event Queue thread by first invoking the static initialization method of the AWT Delegate on the agi.core.awt.AgAWT_JNI class. An example would be:

```
AgAwt_JNI.initialize_AwtDelegate();
```

6) Find the "TODO: #9" comment. Underneath this comment, initialize the STK Java API's Java Native Interface (JNI) connection to the STK Java API's JNI Dll's(or so's) for Custom Application types (using agi.stkengine.AgStkCustomApplication_JNI class static method). Also pass a parameter of true to the initialization method in order to utilize the "Auto Class Cast" feature added in STK 10.0 that provides standard Java casting capabilities from base to derived types where possible. Note in previous releases only "constructor based casting" was available, which is when you pass a base type such as IAgVePropagator into the constructor of AgVePropagatorTwoBody in order to create an instance. An example would be:

```
AgStkCustomApplication_JNI.initialize(true);
```

Note in some cases it may still be necessary to fall back to the constructor based casting when runtime casting exceptions occur during development of your application. For more details refer to the "What's New 10.0" Java documentation in the "<install>/help/integration.chm" file.

Migration Note: If you are migrating your code from a previous version of STK Java API, your code should still work but you will be causing additional object construction and marshaling between Java and Native code to occur, thus possibly degrading performance. If you wish to not change all of your application code to the new casting mechanism, but rather continue to use the older constructor based casting mechanism, merely pass a parameter value of "false" to the above initialization method call in order to bypass possible performance degradation.

7) Find the "TODO: #10" comment. Underneath this comment, initialize the STK Java API's AWT Components like the Globe, Map and Gfx Analysis visualization controls. An example would be:

```
AgAwt_JNI.initialize_AwtComponents();
```

8) Find the "TODO: #11" comment. Underneath this comment, instantiate an AgSTKXApplicationClass class and set the Main class's data member called m_AgSTKXApplicationClass as a reference to it. An example would be:

```
this.m_AgSTKXApplicationClass = new AgSTKXApplicationClass();
```

9) Find the "TODO: #12" comment. Underneath this comment, instantiate an AgGlobeCntrlClass class and set the Main class's data member called m_AgGlobeCntrlClass as a reference to it. An example would be:

```
this.m_AgGlobeCntrlClass = new AgGlobeCntrlClass();
```

10) Find the "TODO: #13" comment. Underneath this comment, add the m_AgGlobeCntrlClass data member variable for the Globe Control as a component of the centerJPanel JPanel instance. An example would be:

```
centerJPanel.add(this.m_AgGlobeCntrlClass);
```

11) Find the "TODO: #14" comment. Underneath this comment, instantiate an AgMapCntrlClass class and set the Main class's data member called m_AgMapCntrlClass as a reference to it. An example would be:

```
this.m_AgMapCntrlClass = new AgMapCntrlClass();
```

12) Find the "TODO: #15" comment. Underneath this comment, add the m_AgMapCntrlClass data member variable for the Map Control as a component of the centerJPanel JPanel instance. An example would be:

```
centerJPanel.add(this.m_AgMapCntrlClass);
```

13) Find the "TODO: #16" comment. Underneath this comment, enter the Connect command necessary to create an empty default scenario as the parameter to the executeCommand method on the data member variable instance of the AgSTKXApplicationClass. An example would be:

```
this.m_AgSTKXApplicationClass.executeCommand("New / Scenario anyname");
```

14) Find the "TODO: #17" comment. Underneath this comment, enter the Connect command necessary to unload any current scenario loaded as the parameter to the executeCommand method on the data member variable instance of the AgSTKXApplicationClass. An example would be:

```
this.m_AgSTKXApplicationClass.executeCommand("Unload / *");
```

15) Find the "TODO: #18" comment. Underneath this comment, issue the "zoom in" method call on the AgMapCntrlClass data member instance. Refer to the STK Java API javadocs or Eclipse intellisense if you have properly configured your samples via Eclipse User Libraries that are highlighted in the STK Java API documentation in the Integration Developer Kit windows help file. An example would be:

```
this.m_AgMapCntrlClass.zoomIn();
```

16) Find the "TODO: #19" comment. Underneath this comment, issue the "zoom out" method call on the AgMapCntrlClass data member instance. Refer to the STK Java API javadocs or Eclipse intellisense if you have properly configured your samples via Eclipse User Libraries that are highlighted in the STK Java API documentation in the Integration Developer Kit windows help file. An example would be:

```
this.m_AgMapCntrlClass.zoomOut();
```

17) Find the "TODO: #20" comment. Underneath this comment, call the dispose method on any data member instance of AGI Globe, Map, or Gfx Analysis controls you have used

within your application. Failure to do so may cause your application to hang if using the DISPOSE_ON_CLOSE close operation with a JFrame. An example would be:

```
Main.this.m_AgGlobeCntrlClass.dispose();
Main.this.m_AgMapCntrlClass.dispose();
```

18) Find the "TODO: #21" comment. Underneath this comment, uninitialize the STK Java API's AWT Components like the Globe, Map and Gfx Analysis visualization controls. An example would be:

```
AgAwt_JNI.uninitialize_AwtComponents();
```

19) Find the "TODO: #22" comment. Underneath this comment, uninitialize the STK Java API's Java Native Interface (JNI) connection to the STK Java API's JNI Dll's(or so's) for Custom Application types (using agi.stkengine.AgStkCustomApplication_JNI class static method). An example would be:

```
AgStkCustomApplication_JNI.uninitialize();
```

20) Find the "TODO: #23" comment. Underneath this comment, uninitialize the STK Java API's Java Native Interface (JNI) connection for use on the AWT/Swing Event Queue thread by invoking the static uninitialization method of the AWT Delegate on the agi.core.awt.AgAWT_JNI class. An example would be:

```
AgAwt_JNI.uninitialize_AwtDelegate();
```

Respond to events raised by STK X

1) Find the "TODO: #24" comment. Underneath this comment, add an event listener to the m_AgSTKXApplicationClass data member instance and receive and report only New Scenario and Close Scenario events. An example would be:

```
this.m_AgSTKXApplicationClass.addIAgSTKXApplicationEvents2(
new IAgSTKXApplicationEvents2()
{
  public void onAgSTKXApplicationEvent(AgSTKXApplicationEvent e)
  {
    // Catch all exceptions from events, it is poor java coding
    // to pass the exception back to the event dispatcher (JNI),
    // it can cause JVM crashes.
    try
    {
       int type = e.getType();
       if(type == AgSTKXApplicationEvent.TYPE_ON_SCENARIO_NEW)
       {
         String path = (String)e.getParams()[0];
         Main.this.m_EventsJLabel.setText("Scenario created: "+ path);
       }
       else if(type == AgSTKXApplicationEvent.TYPE_ON_SCENARIO_CLOSE)
       {
         String path = (String)e.getParams()[0];
         Main.this.m_EventsJLabel.setText("Scenario closed: "+ path);
```

```
      }
     }
     catch(Throwable t)
     {
      JOptionPane.showMessageDialog(Main.this, t.getMessage());
     }
   }});
```

2) Find the "TODO: #25" comment. Underneath this comment, add an event listener to the m_AgGlobeCntrlClass data member instance and receive only Mouse Move events to determine the "picked" Latitude, Longitude, and Altitude. An example would be:

```
this.m_AgGlobeCntrlClass.addIAgGlobeCntrlEvents(
new IAgGlobeCntrlEvents()
{
  public void onAgGlobeCntrlEvent(AgGlobeCntrlEvent e)
  {
    // Catch all exceptions from events, it is poor java coding
    // to pass the exception back to the event dispatcher (JNI),
   // it can cause JVM crashes.
    try
    {
       int type = e.getType();
       if(type == AgGlobeCntrlEvent.TYPE_MOUSE_MOVE)
       {
         Object[] params = e.getParams();

         Integer x = (Integer)params[2];
         Integer y = (Integer)params[3];

         int xi = x.intValue();
         int yi = y.intValue();

         IAgPickInfoData pickInfoData = null;
         pickInfoData = Main.this.m_AgGlobeCntrlClass.pickInfo(xi,yi);

         if(pickInfoData.getIsLatLonAltValid())
         {
           double lat = pickInfoData.getLat();
           double lon = pickInfoData.getLon();
           double alt = pickInfoData.getAlt();
           String text = "Globe LLA: "+lat+", "+lon+", "+alt;
           Main.this.m_EventsJLabel.setText(text);
         }
       }
     }
     catch(Throwable t)
     {
       JOptionPane.showMessageDialog(Main.this, t.getMessage());
     }
  }});
```

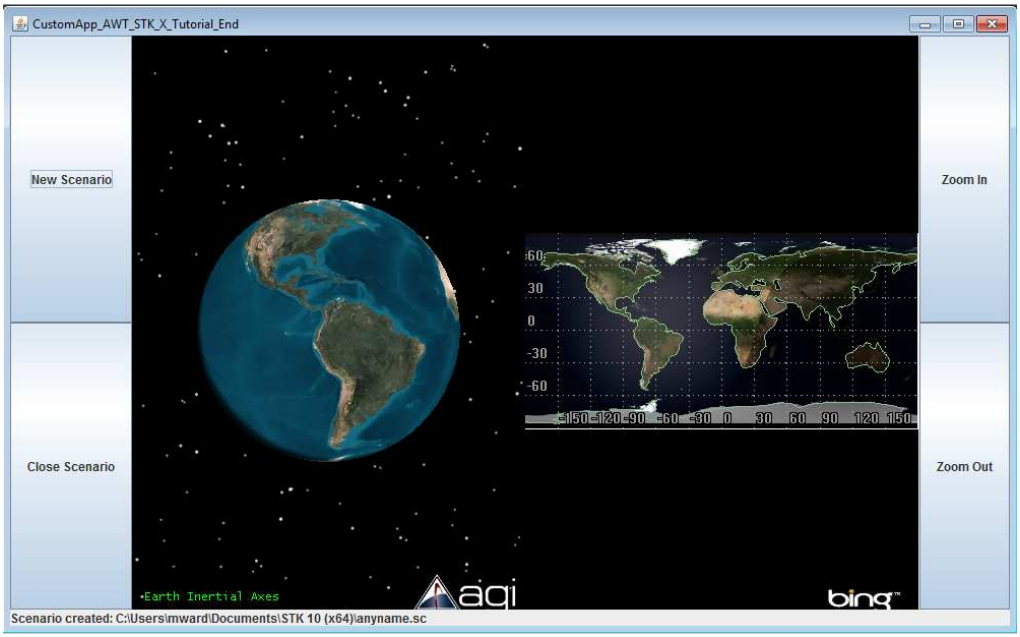
3) Find the "TODO: #26" comment. Underneath this comment, add an event listener to the m_AMapCntrlClass data member instance and receive only Mouse Move events to determine the "picked" Latitude, Longitude, and Altitude. An example would be:

```
this.m_AgMapCntrlClass.addIAgMapCntrlEvents(
new IAgMapCntrlEvents()
{
  public void onAgMapCntrlEvent(AgMapCntrlEvent e)
  {
    // Catch all exceptions from events, it is poor java coding
    // to pass the exception back to the event dispatcher (JNI),
   // it can cause JVM crashes.
    try
    {
       int type = e.getType();
       if(type == AgMapCntrlEvent.TYPE_MOUSE_MOVE)
       {
         Object[] params = e.getParams();

         Integer x = (Integer)params[2];
         Integer y = (Integer)params[3];

         int xi = x.intValue();
         int yi = y.intValue();

         IAgPickInfoData pickInfoData = null;
         pickInfoData = Main.this.m_AgMapCntrlClass.pickInfo(xi,yi);

         if(pickInfoData.getIsLatLonAltValid())
         {
           double lat = pickInfoData.getLat();
           double lon = pickInfoData.getLon();
           double alt = pickInfoData.getAlt();
           String text = "Map LLA: "+lat+", "+lon+", "+alt;
           Main.this.m_EventsJLabel.setText(text);
         }
       }
     }
     catch(Throwable t)
     {
       JOptionPane.showMessageDialog(Main.this, t.getMessage());
     }
  }});
```

Compiling and running the application

1) At this point, you can compile and run your application. It should look something like this:

2) Click the New Scenario button. You should see an application like the following:



3) Click the Zoom In button. Use the mouse to define the zoom area in the Map control as seen above. You can zoom out by clicking the Zoom Out button.