## 16.4 Quality control

**The Quality Control is an interface that may be implemented by those JMF classes such as Codecs that want to offer control to the user on the quality of the output generated by it. The quality is specified by a data of type float. The value of quality ranges from 0.0 signifying minimum quality to 1.0 signifying the maximum quality. Quality control can be used to fix the tradeoff between the quality of the media and the processing power required to get that quality. For example setting the quality factor to a lower value of an encoder implies lower quality, demanding lower processing power and may be lower bandwidth. Figure 16.5 shows the definition of the interface QualityControl.**

| <<interface>> javax.media.control.QualityControl |
|---|
| |
| public float getQuality()<br>public float setQuality(float newQuality)<br>public float getPreferredQuality()<br>public boolean isTemporalSpatialTradeoffSupported() |

**Figure 16.5 The interface QualityControl**

**The example qualityControlDemo.java shown below illustrates how the quality of a JPEG video encoder can be controlled.**

```
/*  The program qualityControlDemo.java  illustrates the use of quality control in making a
suitable tradeoff between the media quailty and the amount of processing required to encode
the media or  the bandwidth occupied by the media.
*/
import JMFUtil.*;
import javax.media.*;
import javax.media.protocol.*;
import java.io.*;
import javax.media.format.*;
import java.util.*;
import javax.media.control.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class qualityControlDemo {

DataSource ds = null;
Processor processor = null;
java.awt.Component QCcomp;

public  qualityControlDemo() {

        MediaLocator ml = captureDev.getVideoML();
        try{
```

```
                processor = Manager.createProcessor( ml );

        } catch (NoProcessorException npe  ) {
                 System.out.println( "Couldn't create processor : " + npe );
        } catch (IOException ioe) {
              System.out.println( "IOException creating processor : " + ioe);
        }

        procUtil util = new procUtil(processor);

        boolean result = util.syncConfigure();
        if (result == false)
        {   System.out.println( "Couldn't configure processor \n" );
                System.exit(0);
        }

        processor.setContentDescriptor( null );

        VideoFormat vFormat = new VideoFormat( VideoFormat.JPEG);
        Format setFormat = util.setVideoFormat(vFormat);
        System.out.println( "The video format set is :" + setFormat );

        result = util.syncRealize();
        if (result == false)
        {   System.out.println( "Couldn't realize processor \n " );
            System.exit(0);
        }

       QualityControl  qc = (QualityControl) processor.getControl
                                        ("javax.media.control.QualityControl");
      if( qc != null )
        {
                float prequality = qc.getPreferredQuality();
                System.out.println("Encoder preferred quality is :"   +prequality);
                float quality = qc.setQuality(0.3f);
                System.out.println("Video Quality value set to  :  "+quality);
               QCcomp = qc.getControlComponent();
        }

        JFrame frame = util.displayGUI();
        frame.setTitle("Quality Control Demo");
        if( QCcomp != null)
        {
              frame.getContentPane().add( QCcomp, BorderLayout.NORTH);
        }
        Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
        frame.setLocation(  d.width/2, d.height/2);
        frame.setVisible( true);
        processor.start();
}

public static void main( String args[]) {
```

```
        qualityControlDemo proc = new qualityControlDemo();
}

}
```

---

**The explanation of the example qualityControlDemo is as follows. By using the captureDev utility we get the MediaLocator of the video capture device. We then create a Processor for the video DataSource using the Manager. The Processor is configured and realized using the procUtil utility. The video Format is set to JPEG format. We check whether the Processor offers the QualityControl using the method,**

```
QualityControl  qc = (QualityControl) processor.getControl
                                        ("javax.media.control.QualityControl");
```

**A null is returned if QualityControl is not offered by the Processor. We assume that this QualityControl if offered by the Processor corresponds to the QualityControl of the video Codec. We set the quality to 0.3. As it is true for all objects implementing the control interface, the QualityControl object may offer a GUI to let the user to exercise the control. We check whether a Qualitycontrol component is offered by the specific encoder we are using. If so the QualityControl Component is presented to the user along with the visual and the controlPanel Component of the Processor.**


**Figure 16.6 The screen shot of QualityControlDemo.java**