Captura de medios con la JMF

Servidores de Información Multimedia
2º Ingeniero Técnico de Telecomunicación – Imagen y Sonido

Departamento de Ingeniería Telemática
Universidad Carlos III de Madrid

2 Índice

- ¿Qué es un dispositivo de captura?
- Proceso de captura de medios
  - Accediendo a los dispositivos de captura
  - Creando un procesador o un reproductor
- Controles de un dispositivo de captura
- Guardando en disco el contenido capturado
- Ejemplos

3 ¿Qué es un dispositivo de captura?

- No es más que un DataSource (PullDataSource, PullBufferDataSource, PushDataSource o PushBufferDataSource) que además implementa la interfaz CaptureDevice.
- Como todo DataSource tiene una serie SourceStreams asociados a través de los que se accede a los medios capturados.
- El DataSource del dispositivo de captura puede usarse para alimentar un Player o un Processor en función de si se quiere presentar (render) en tiempo de captura el medio que se está capturando o si se quiere procesar, almacenar o retransmitir por la red.

4 Proceso de captura de medios

- Para capturar medios con la JMF básicamente se siguen los siguientes pasos:
  - Localizar el dispositivo de captura que queremos usar preguntando al CaptureDeviceManager.
  - Obtener el objeto de tipo CaptureDeviceInfo para el dispositivo.
  - Obtener el MediaLocator del objeto CaptureDeviceInfo y usarlo para crear el DataSource.
  - Crear un Player o Processor usando el DataSource.
  - Arrancar el Player o Processor para iniciar el proceso de captura.
- Si capturamos el medio a través de un reproductor (Player) solamente podremos presentarlo (pantalla o altavoces).
- Veamos a continuación más detalles del proceso de captura.

5 Accediendo a los dispositivos de captura

- Se accede a través del CaptureDeviceManager que es una especie de registro donde se encuentran los capturadores disponibles en la JMF.
- Para obtener la lista de capturadores disponibles se ejecuta el método CaptureDeviceManager.getDeviceList al que se le pasa el formato del medio que queremos capturar y que nos devuelve un Vector cuyos componentes son de tipo CaptureDeviceInfo.
- Otra posibilidad de obtener el CaptureDeviceInfo de un capturador es mediante la invocación del método CaptureDeviceManager.getDevice al que se le pasa como String el nombre del dispositivo de captura que queremos.
- A modo de ejemplo:
  - CaptureDeviceInfo deviceInfo = CaptureDeviceManager.getDevice("deviceName");

6 Creando un procesador o un reproductor

- Una vez que se tiene el CaptureDeviceInfo del capturador a usar se invoca el método: CaptureDeviceInfo.getLocator() para obtener el MediaLocator.
- Una vez tenemos el MediaLocator lo podemos usar directamente para crear el Player o Processor o bien lo podemos usar para crear un DataSource que luego usaremos para crear el Player o Processor.
- Para iniciar la captura se llama al método start del Player o Processor.

7 Controles de un dispositivo de captura

- Los dispositivos de captura normalmente implementan dos controles: PortControl y MonitorControl.
- Estos controles se pueden obtener a través del método getControl (pasándole el nombre del control que queremos obtener) o getControls (para obtener todos los controles) del DataSource que se ha creado para el dispositivo de captura.
- El PortControl proporciona un mecanismo para seleccionar el puerto del cual se quiere capturar los datos (line-in, microfono…).
- El MonitorControl proporciona un mecanismo para permitir la monitorización de la captura.
- Como el resto de controles, si tienen un componente visual se puede recuperar con el método getControlComponent.

8 Guardando en disco el contenido capturado

- Una vez asociado un Processor al dispositivo de captura
  - Obtener el DataSource de salida del Processor llamando getDataOutput.
  - Construir un DataSink para escribir en disco llamando a Manager.createDataSink. Hay que pasarle el DataSource del Processor y el MediaLocator que especifica el fichero donde se quiere escribir.
  - Invocar open en el DataSink para abrir el fichero.
  - Invocar start en el DataSink.
  - Invocar start en el Processor para empezar la captura de datos.
  - Esperar a que llegue un EndOfMediaEvent (que lanzan en general los Controller), o cualquier otro evento de tiempo de medios o de usuario.
  - Invocar stop en el Processor para finalizar la captura.
  - Invocar close en el Processor.
  - Cuando se cierra el Processor y el DataSink lanza un EndOfStreamEvent, invocar close en el DataSink.

9 Guardando en disco el contenido capturado (cont)

- Veamos un ejemplo de código para manejar la escritura en disco de la captura de un medio:

```
DataSink sink;
MediaLocator dest = new MediaLocator("file://newfile.wav");
try {
        sink = Manager.createDataSink(p.getDataOutput(), dest);
        sink.open();
        sink.start();
} catch (Exception) {}
```


10 Ejemplo 1: capturando audio de un micrófono

- Veamos el siguiente ejemplo en el que capturamos audio de un micrófono y lo renderizamos a los altavoces del sistema mediante un Player:

```
// Get the CaptureDeviceInfo for the live audio capture device
Vector deviceList = CaptureDeviceManager.getDeviceList(new
                        AudioFormat("linear", 44100, 16, 2));
if (deviceList.size() > 0)
        di = (CaptureDeviceInfo)deviceList.firstElement();
else
// Exit if we can't find a device that does linear, 44100Hz, 16 bit,
// stereo audio.
    System.exit(-1);

// Create a Player for the capture device:
try{
    Player p = Manager.createPlayer(di.getLocator());
} catch (IOException e) {
} catch (NoPlayerException e) {}
```

11 Ejemplo 2: Grabando en disco la captura

```
CaptureDeviceInfo di = null;
Processor p = null;
StateHelper sh = null;
Vector deviceList = CaptureDeviceManager.getDeviceList(new
                        AudioFormat(AudioFormat.LINEAR, 44100, 16, 2));
if (deviceList.size() > 0)
    di = (CaptureDeviceInfo)deviceList.firstElement();
else
    // Exit if we can't find a device that does linear,
 // 44100Hz, 16 bit,
    // stereo audio.
    System.exit(-1);
try {
    p = Manager.createProcessor(di.getLocator());
    sh = new StateHelper(p);
} catch (IOException e) {
    System.exit(-1);
} catch (NoProcessorException e) {
    System.exit(-1);
}
```


12 Ejemplo 2: Grabando en disco la captura (cont)

```
// Configure the processor
if (!sh.configure(10000))
    System.exit(-1);
// Set the output content type and realize the processor
p.setContentDescriptor(new
          FileTypeDescriptor(FileTypeDescriptor.WAVE));
if (!sh.realize(10000))
    System.exit(-1);
// get the output of the processor
DataSource source = p.getDataOutput();
// create a File protocol MediaLocator with the location of the
// file to which the data is to be written
MediaLocator dest = new MediaLocator("file://foo.wav");
// create a datasink to do the file writing & open the sink to
// make sure we can write to it.
DataSink filewriter = null;
try {
    filewriter = Manager.createDataSink(source, dest);
    filewriter.open();
} catch (NoDataSinkException e) {
    System.exit(-1);
} catch (IOException e) {
    System.exit(-1);
} catch (SecurityException e) {
    System.exit(-1);
}
```

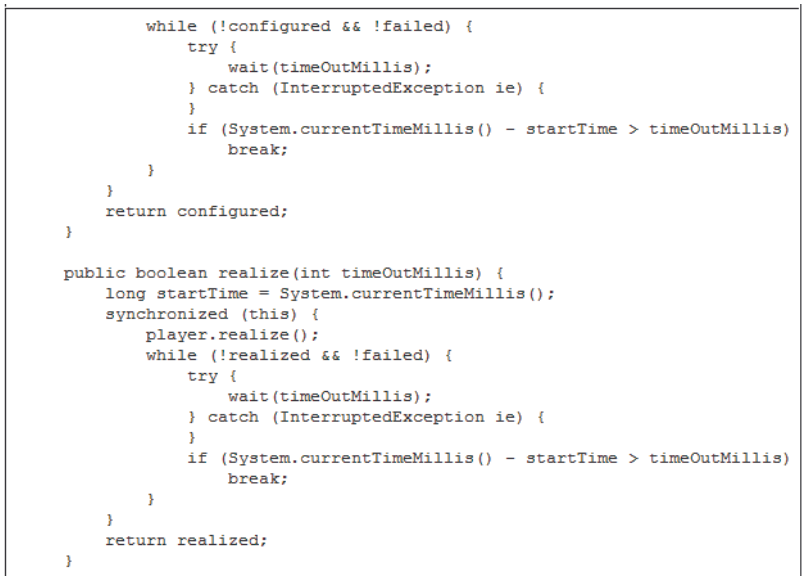
13 Ejemplo 2: Grabando en disco la captura (cont)

```
        // if the Processor implements StreamWriterControl, we can
        // call setStreamSizeLimit
        // to set a limit on the size of the file that is written.
        StreamWriterControl swc = (StreamWriterControl)
            p.getControl("javax.media.control.StreamWriterControl");
        //set limit to 5MB
        if (swc != null)
            swc.setStreamSizeLimit(5000000);

        // now start the filewriter and processor
        try {
            filewriter.start();
        } catch (IOException e) {
            System.exit(-1);
        }
        // Capture for 5 seconds
        sh.playToEndOfMedia(5000);
        sh.close();
        // Wait for an EndOfStream from the DataSink and close it...
        filewriter.close();
```


14 La clase de soporte StateHelper (I)

```
import javax.media.*;


public class StateHelper implements javax.media.ControllerListener {

    Player player = null;
    boolean configured = false;
    boolean realized = false;
    boolean prefetched = false;
    boolean eom = false;
    boolean failed = false;
    boolean closed = false;

    public StateHelper(Player p) {
        player = p;
        p.addControllerListener(this);
    }

    public boolean configure(int timeOutMillis) {
        long startTime = System.currentTimeMillis();
        synchronized (this) {
            if (player instanceof Processor)
                ((Processor)player).configure();
            else
                return false;
```

15 La clase de soporte StateHelper (II)

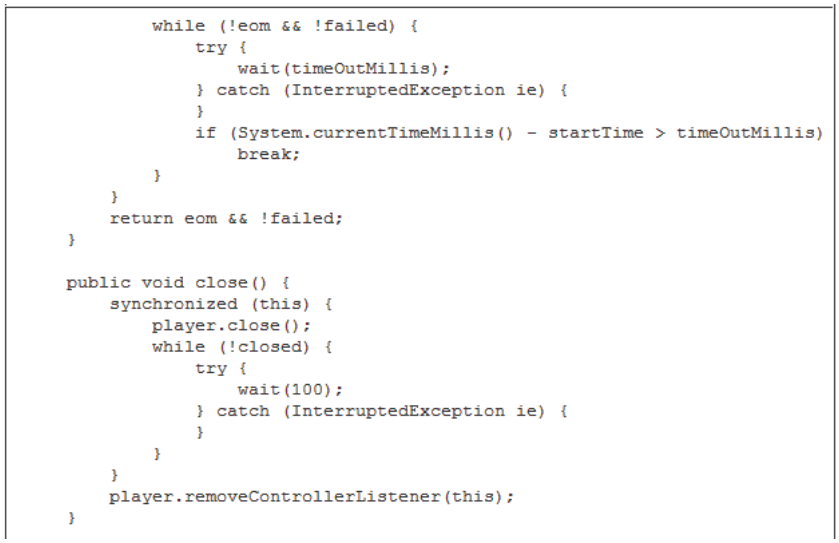


```
            while (!configured && !failed) {
                try {
                    wait(timeOutMillis);
                } catch (InterruptedException ie) {
                }
                if (System.currentTimeMillis() - startTime > timeOutMillis)
                    break;
            }
        }
        return configured;
    }

    public boolean realize(int timeOutMillis) {
        long startTime = System.currentTimeMillis();
        synchronized (this) {
            player.realize();
            while (!realized && !failed) {
                try {
                    wait(timeOutMillis);
                } catch (InterruptedException ie) {
                }
                if (System.currentTimeMillis() - startTime > timeOutMillis)
                    break;
            }
        }
        return realized;
    }
```


16 La clase de soporte StateHelper (III)

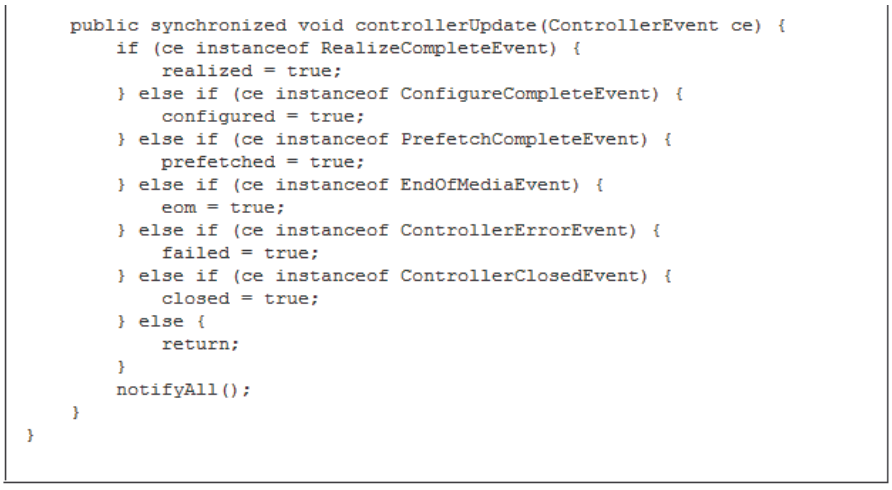
```
    public boolean prefetch(int timeOutMillis) {
        long startTime = System.currentTimeMillis();
        synchronized (this) {
            player.prefetch();
            while (!prefetched && !failed) {
                try {
                    wait(timeOutMillis);
                } catch (InterruptedException ie) {
                }
                if (System.currentTimeMillis() - startTime > timeOutMillis)
                    break;
            }
        }
        return prefetched && !failed;
    }
    public boolean playToEndOfMedia(int timeOutMillis) {
        long startTime = System.currentTimeMillis();
        eom = false;
        synchronized (this) {
            player.start();
```

17 La clase de soporte StateHelper (IV)



```
            while (!eom && !failed) {
                try {
                    wait(timeOutMillis);
                } catch (InterruptedException ie) {
                }
                if (System.currentTimeMillis() - startTime > timeOutMillis)
                    break;
            }
        }
        return eom && !failed;
    }

    public void close() {
        synchronized (this) {
            player.close();
            while (!closed) {
                try {
                    wait(100);
                } catch (InterruptedException ie) {
                }
            }
        }
        player.removeControllerListener(this);
    }
```


18 La clase de soporte StateHelper (V)



```
    public synchronized void controllerUpdate(ControllerEvent ce) {
        if (ce instanceof RealizeCompleteEvent) {
            realized = true;
        } else if (ce instanceof ConfigureCompleteEvent) {
            configured = true;
        } else if (ce instanceof PrefetchCompleteEvent) {
            prefetched = true;
        } else if (ce instanceof EndOfMediaEvent) {
            eom = true;
        } else if (ce instanceof ControllerErrorEvent) {
            failed = true;
        } else if (ce instanceof ControllerClosedEvent) {
            closed = true;
        } else {
            return;
        }
        notifyAll();
    }
}
```

19 Ejemplo 3:Codificación del audio capturado

```
// Configure the processor
if (!sh.configure(10000))
    System.exit(-1);
// Set the output content type
p.setContentDescriptor(new
        FileTypeDescriptor(FileTypeDescriptor.WAVE));

// Get the track control objects
TrackControl track[] = p.getTrackControls();
boolean encodingPossible = false;
// Go through the tracks and try to program one of them
// to output ima4 data.
for (int i = 0; i < track.length; i++) {
    try {
        track[i].setFormat(new AudioFormat(AudioFormat.IMA4_MS));
        encodingPossible = true;
    } catch (Exception e) {
        // cannot convert to ima4
        track[i].setEnabled(false);
    }
}

if (!encodingPossible) {
    sh.close();
    System.exit(-1);
}
// Realize the processor
if (!sh.realize(10000))
    System.exit(-1);
```


20 Ejemplo 4: Creación del Procesador con un ProcessorModel

```
Format formats[] = new Format[2];
formats[0] = new AudioFormat(AudioFormat.IMA4);
formats[1] = new VideoFormat(VideoFormat.CINEPAK);
FileTypeDescriptor outputType =
    new FileTypeDescriptor(FileTypeDescriptor.QUICKTIME);
Processor p = null;

try {
    p = Manager.createRealizedProcessor(new ProcessorModel(formats,
                                                    outputType));
} catch (IOException e) {
    System.exit(-1);
} catch (NoProcessorException e) {
    System.exit(-1);
} catch (CannotRealizeException e) {
    System.exit(-1);
}
// get the output of the processor
DataSource source = p.getDataOutput();
// create a File protocol MediaLocator with the location
// of the file to
// which bits are to be written
MediaLocator dest = new MediaLocator("file://foo.mov");
// create a datasink to do the file writing & open the
// sink to make sure
// we can write to it.
DataSink filewriter = null;
```

21 Ejemplo 4: Creación del Procesador con un ProcessorModel

```
try {
    filewriter = Manager.createDataSink(source, dest);
    filewriter.open();
} catch (NoDataSinkException e) {
    System.exit(-1);
} catch (IOException e) {
    System.exit(-1);
} catch (SecurityException e) {
    System.exit(-1);
}
// now start the filewriter and processor
try {
    filewriter.start();
} catch (IOException e) {
    System.exit(-1);
}
p.start();
// stop and close the processor when done capturing...
// close the datasink when EndOfStream event is received...
```


22 Cuestiones de refuerzo

- ¿Qué elemento de la arquitectura de JMF se utiliza para modelar un dispositivo de captura: **`DataSource`**, **`CaptureSource`**, **`Player`**, **`DataSink`** o **`Processor`**?
- ¿Qué interfaz diferencia a un dispositivo de captura de otros elementos: **`CaptureDevice`**, **`CaptureStream`** o **`Capture`**?
- Explicitar los cinco pasos básicos que se siguen a la hora de capturar un medio
- ¿Cuál es la función básica de **`CaptureDeviceManager`** a la hora de realizar una captura?
- ¿Cómo obtenemos el objecto **`MediaLocator`** necesario para capturar un medio?
- ¿Con qué método se inicia la captura del medio?
- ¿Qué dos tipos de controles implementan los dispositivos de captura normalmente? ¿Para qué sirven cada uno de ellos?
- Indique los pasos necesarios para almacenar un medio de captura en un fichero.

23 Cuestiones de refuerzo

- En el código de la figura 9, indicar que pasa si no se llama al método `start()`
- En el ejemplo de la transparencia 10, diga que pasa con el audio capturado, se presenta por los altavoces o por el contrario se almacena en disco.
- En el ejemplo de las transparencias 11, 12, 13, 14, 15, 16, 17 y 18 identifique los principales pasos que han sido dados en el código.

24 Autoría

- Mario Muñoz Organero
- Pablo Basanta Val

  +Material de refuerzo (2009)