

Programmer la vidéo

Jean-Marc Farinone
(Maître de Conférences CNAM)

Plan de l'exposé

- Démonstrations, Présentation, Historique
- Lecture Vidéo
- Capture Vidéo
- Bibliographie

Java Media Framework (JMF) : Démonstrations, Présentation, Historique

Présentation

- On peut lire divers formats vidéos à l'aide de Java Media Framework (JMF) depuis JMF 1.0 dans une application indépendante ou dans une applet.
- On peut de plus, capturer, sauvegarder, transmettre, transcoder de la vidéo depuis JMF 2.0
- version actuelle JMF 2.1.1e (en avril 2010) depuis pas mal de temps

Médias supportés par JMF

- JMF traite les données utilisant le temps.
Les divers types traités par JMF sont :
 - protocoles: FILE, HTTP, FTP, RTP
 - audio: AIFF, AU, AVI, GSM, MIDI, MP2, MP3, QT, RMF, WA
 - vidéo: AVI, MPEG-1, QT, H.261, H.263
 - autres : Flash 2, HotMedia

JMF : historique

- JMF développé par Sun Microsystems, Silicon Graphics, Intel, IBM et RealNetworks est composé de trois parties : Player, Capture, Conferencing.
- Début des spécifications en 1996. Première implémentation (version 0.95) rendue publique en Février 1997.
- Les implémentations "natives" font appel aux couches logicielles natives multimédia de la plateforme
- Il existe une implémentation 100% pur Java (cross-platform)

La "pile" "native" JMF pour la vidéo

Notre programme

Classes Java JMF

Bibliothèques natives pour la vidéo

CPU + entrée caméra, microphone + sortie HP, écran

JMF : installation

JMF : installation

- Télécharger à :
`http://java.sun.com/products/java-media/jmf/2.1.1/download.html` en choisissant sa plate-forme (Win32, Linux, Solaris ou cross platform)
- Lancer le `.exe` ou le `.sh` ou ouvrir le `.zip`.
- L'installation peut être mise n'importe où (! = Java 3D)
- Voir ensuite les configurations à positionner à : `http://java.sun.com/products/java-media/jmf/2.1.1/setup.html`

JMF : installation (suite)

- CLASSPATH a été positionné de sorte à repérer `jmf.jar` et `sound.jar` du répertoire `lib` téléchargé.
- Ou mieux !!, mettre ces 2 `.jar` dans `%JAVA_HOME%\jre\lib\ext`
- Tester l'install en lisant la page d'URL : <http://java.sun.com/products/java-media/jmf/2.1.1/jmfdiagnostics.html> qui doit renvoyer : [Code Samples and Apps](#)
[JMF - Diagnostics Applet](#)

The applet searches for JMF classes and libraries to see if JMF is setup correctly on your system and if your browser.

If there are error messages, look below the applet for trouble shooting information.

Here is the [Source Code](#) for the JMF Diagnostics Applet. It contains both the java and class files used in the applet.

Note: Internet Explorer might crash if you are using JMF 1.0.2 or an earlier version.



JMF

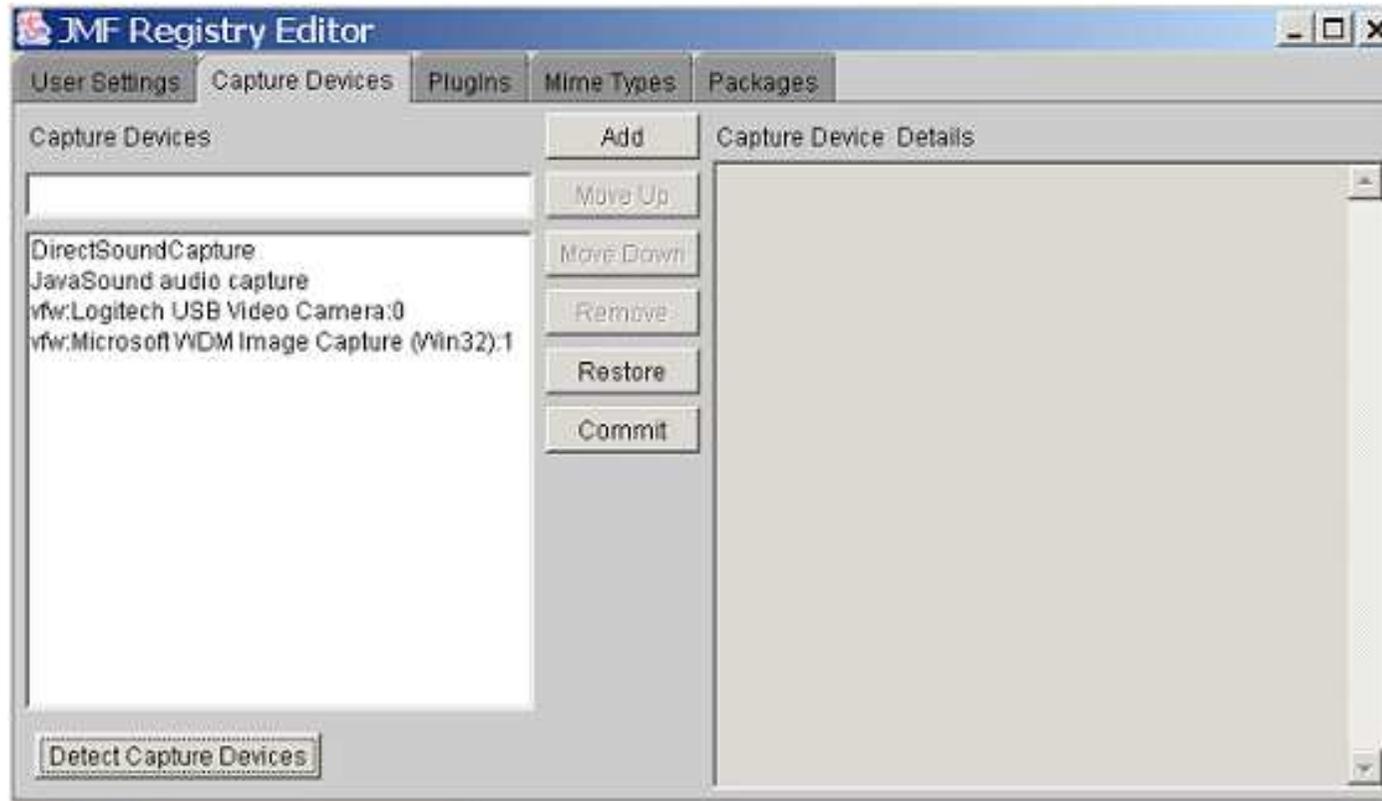
Installation caméra

- Après installation de JMF, plusieurs programmes sont disponibles dont `JMFRegistry`, `JMStudio` (qui possède un raccourci sur le bureau sous Win32).
- Leurs sources sont disponibles à :
`http://java.sun.com/products/java-media/jmf/2.1.1/samples/jmapps-src-211.zip`
- Après avoir installé une caméra, il faut la faire connaître par JMF.
- Lancer `JMFRegistry`

Faire connaître la caméra par JMFRRegistry

- Voir à :
`http://java.sun.com/products/java-media/jmf/2.1.1/jmfregistry/jmfregistry.html`
- Gère tous les ajouts, retraits, etc. de périphériques.
- Pour faire connaître une source vidéo, utiliser l'onglet Capture Device Manager.
- Utiliser les boutons Add puis Commit

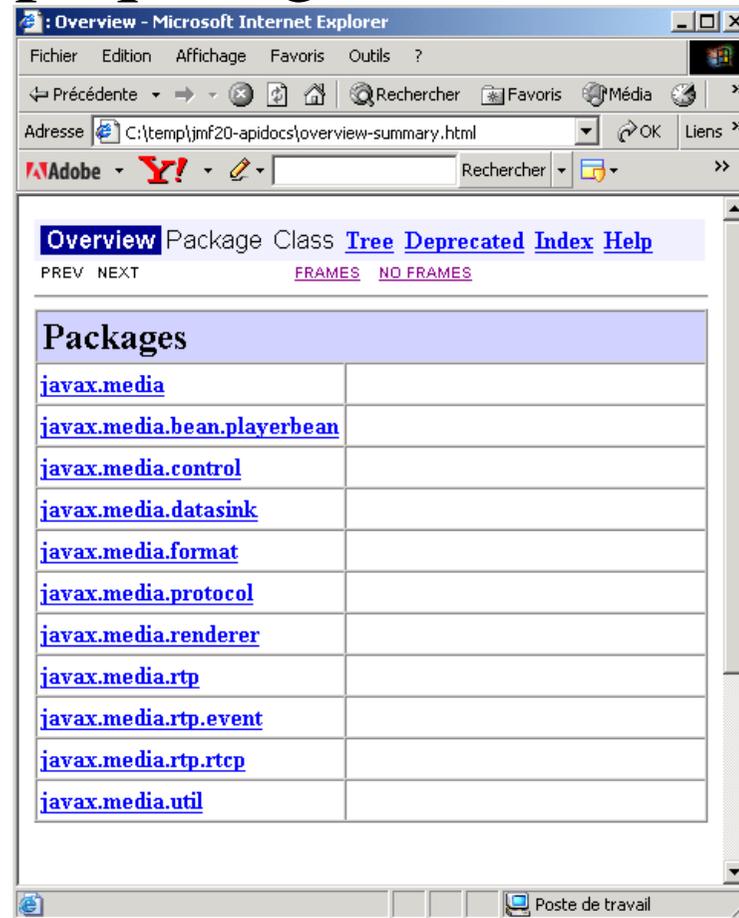
Faire connaître la caméra par JMFRRegistry



JMF : Lecture Vidéo

Lecture Vidéo : architecture

- Un paquetage principal : `javax.media`
- Et 10 sous-paquetages



Lecture Vidéo

- `Player` = lecteur vidéo
- Contrôle le chargement, l'acquisition des ressources multimédia, l'exécution (démarrage, arrêt, vitesse d'exécution, ...) d'un document multimédia.
- Obtenu en demandant au gestionnaire de documents multimédia (le `Manager`) de retourner celui approprié pour gérer la ressource multimédia
- Syntaxe :

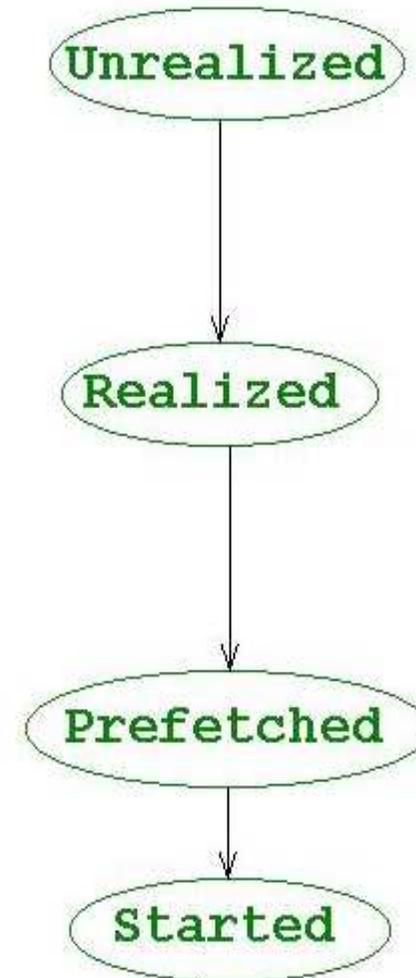
```
Player lePlayer =
```

```
Manager.createPlayer(URLduDocumentMultimedia);
```

JMF

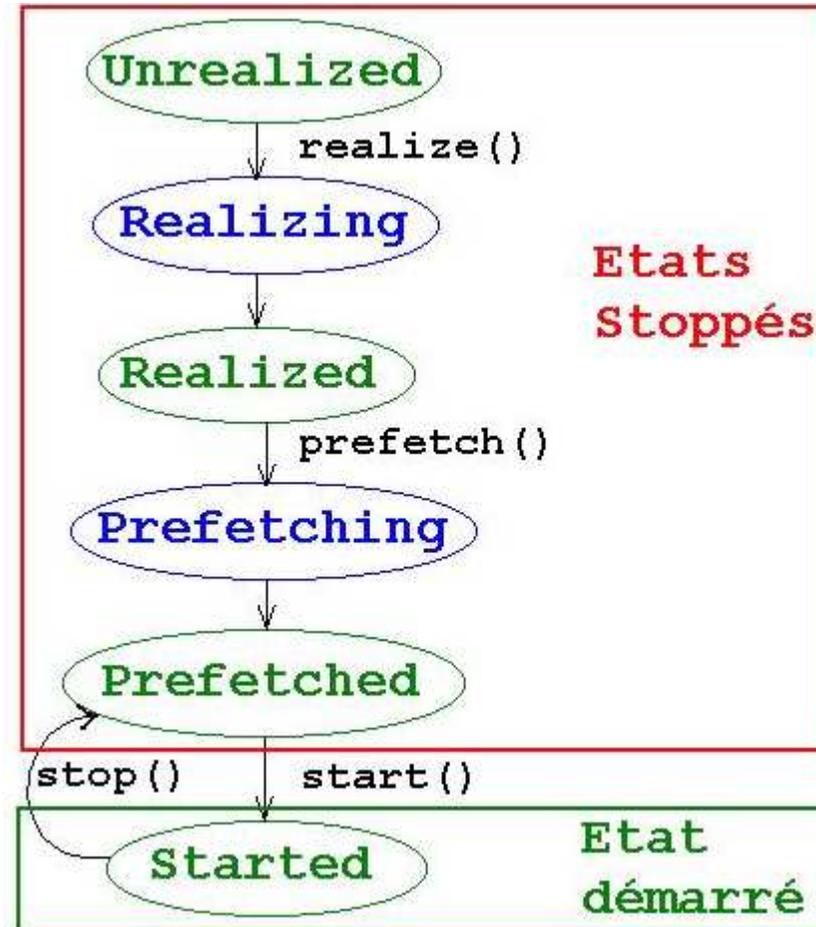
Les états fondamentaux d'un Player

- Les principaux états sont :
- Dans l'état `Unrealized`, aucune ressource n'est attribuée.
- Dans l'état `Realized`, le `Player` sait quelles ressources il doit avoir
- Dans l'état `Prefetched`, il a acquis les ressources
- Dans l'état `Started` l'exécution est en cours.



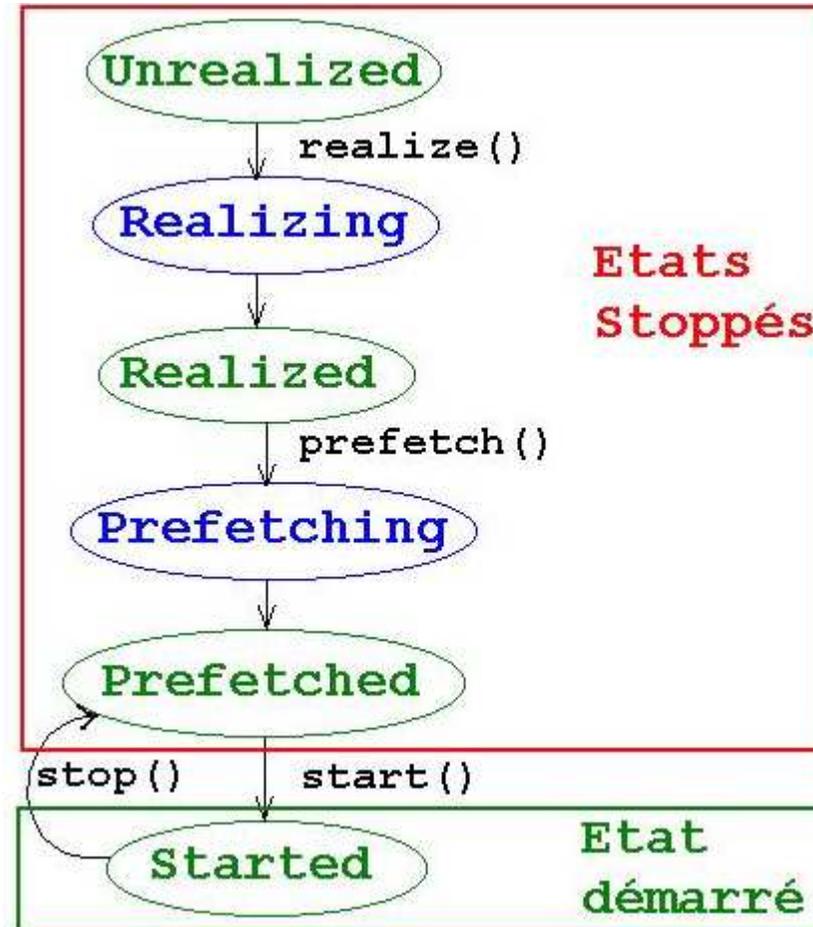
Les états d'un Player (suite)

- Le passage d'un état fondamental à un autre peut prendre du temps, aussi il a été défini des états intermédiaires.



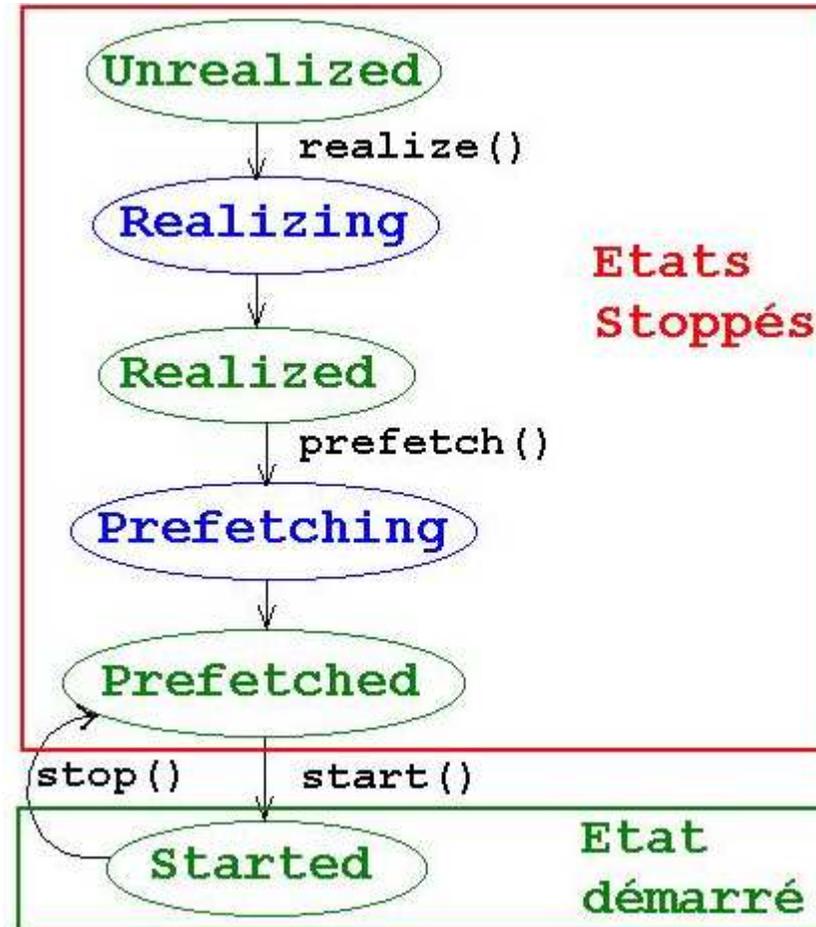
Les états d'un Player (suite)

- Le passage des états `Realizing` à `Realized` et de `Prefetching` à `Prefetched` est automatique et réalisé par le moteur multimédia.



Les états d'un Player (fin)

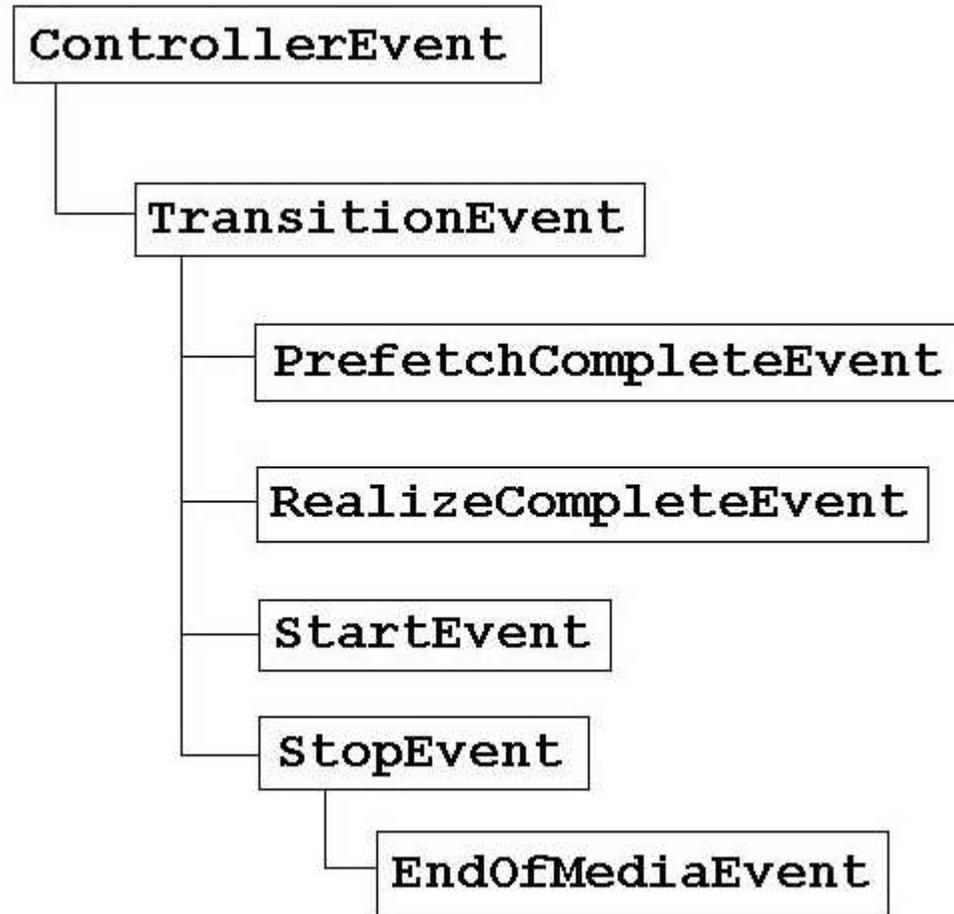
- Le passage entre les autres états peut être effectué par la demande d'exécution de méthodes.



Les événements de transition dans un `Player`

- A chaque changement d'état, un événement (objet d'une sous classe de la classe `ControllerEvent`) est généré par le `Player`.
- Cet événement est envoyé au(x) `ControllerListener(s)` associé(s) au `Player`.
- Les `ControllerListeners` lancent alors leur méthode : `public synchronized void controllerUpdate(ControllerEvent event)`

Arborescence des événements de transition dans un Player



Lecture vidéo : trame

```
import java.awt.*;
import java.net.*;
import javax.media.*;

public class lanceAppliMult extends Frame implements ControllerListener {
    // Le Player
    private Player masterPlayer = null;

    // Le panneau de controle (execution, avance rapide, ...) du Player
    private Component masterControl = null;

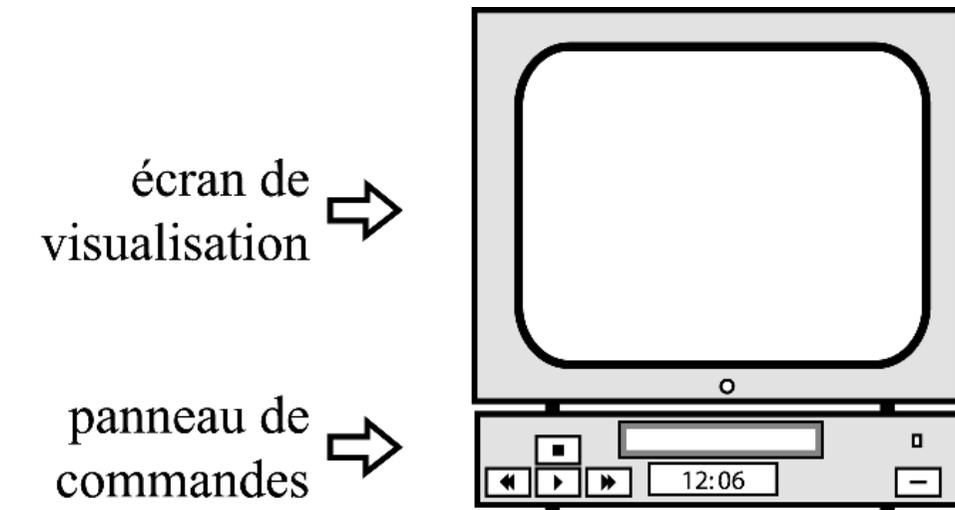
    // Le composant visuel (i.e. l'écran) du Player
    private Component masterVisualComp = null;

    public static void main (String args[]) {
        URL masterURL = null;
        lanceAppliMult app = null;

        masterURL = new URL(args[0]);
        app = new lanceAppliMult(masterURL);
        ...
    }

    public lanceAppliMult(URL masterURL) {
        ...
        masterPlayer = Manager.createPlayer(masterURL);
        masterPlayer.addControllerListener(this);
        masterPlayer.realize();
    }
}
```

Composants graphiques pour la lecture vidéo



- **Écran de visualisation** = Component obtenu :
`masterPlayer.getVisualComponent();`
- **Panneau de commandes** = Component obtenu :
`masterPlayer.getControlPanelComponent();`

Programme de lecture vidéo :

trame (fin)

```
/**
 * Le traitement des evenements video
 *
 * Cette méthode est la methode a implanter, provenant
 * de l'interface javax.media.ControllerListener
 *
 */
public synchronized void controllerUpdate(ControllerEvent evt) {
    if (evt instanceof RealizeCompleteEvent) {
        ...
        masterVisualComp = masterPlayer.getVisualComponent();
        if (masterVisualComp != null) {
            .....add(masterVisualComp);
        }

        masterControl = masterPlayer.getControlPanelComponent();
        if (masterControl != null) {
            .....add(masterControl);
        }
    }
    if (evt instanceof StartEvent) {
        ...
    }
}
}
```

Démonstrations

- Vidéos dans des applications indépendantes

```
1JMFappliAVI.bat
```

```
2JMFappliMOV.bat
```

```
3JMFappliMPG.bat
```

- Remarques :
- C'est le même programme pour les 3 formats
- ... et pour les formats audio (MIDI, RMF, WAV, ...)
- ... en chargement http

```
7JMFPIazollaHttp.bat
```

Conclusion : lecture vidéo

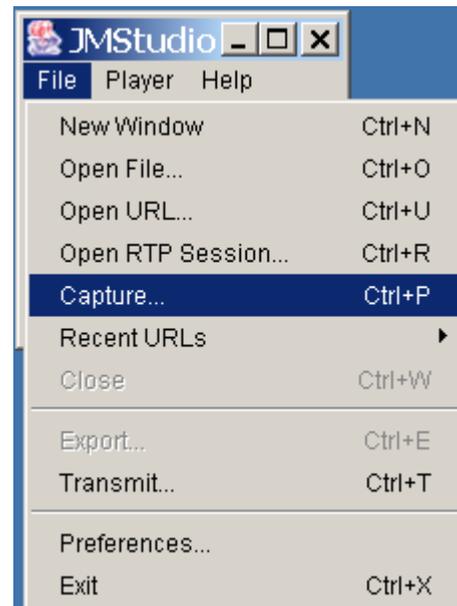
- Vidéos dans des applications indépendantes
- Et dans les applets
- => téléchargement de spots musicaux

JMF : Capture Vidéo

JMStudio

- Page initiale :
`http://java.sun.com/products/java-media/jmf/2.1.1/jmstudio/jmstudio.html`
- Est une application indépendante Java qui exécute, capture (= capable de récupérer ce qui provient du périphérique caméra), transmet et sauvegarde de la vidéo
- Source complet disponible en `.zip` de 211 Ko

Demonstration JMStudio



- Fermer les sorties
HP de
l'ordinateur !

Capture vidéo avec JMStudio

- Lancer JMStudio
- Sélectionner File | Capture
- Indiquer vidéo (préciser la caméra) ou/et capture audio (directe). Cliquez OK.
- Sélectionner File | Export. Sélectionner formats de sortie (avi, mov, ...). Cliquez Save
- Indiquez le fichier résultat. La capture est en cours (voir timer !!). Cliquez Stop pour terminer.

Démonstration capture vidéo avec JMStudio

`8JMStudio.bat`

Lecture avec JMStudio

- Lancer JMStudio
- Sélectionner File | Open File
- Sélectionner le fichier audio vidéo

Démonstration lecture de la capture
précédente

Programme de capture vidéo

- Chrystèle Carré (valeur C IAGL 2002)
- Similaire à JMStudio
- Trois parties :
 - recherche de pilotes
 - affichage de la capture
 - enregistrement de la capture

Capturing video

- Three steps:
 - find audio and video capture devices
 - choose one
 - obtain a video datasource from this device
- Two proposed programs
 - display capture video
 - save capture video

Capturing video

- A demo which shows captured video

9 lanceCapture.bat
première partie

First step: find the devices

- Obtain all the devices by:

```
Vector theDevices =  
CaptureDeviceManager.getDeviceList(null);
```

- Obtain infos for a device in this vector by:

```
CaptureDeviceInfo cdi =  
(CaptureDeviceInfo) theDevices.elementAt(i)
```

- Obtain formats managed by this device by:

```
Format[] formats = cdi.getFormats();
```



Second step: find the DataSources

- The CaptureDeviceInfo audioCDI and videoCDI have been initialized
- The DataSources are found by:

```
DataSource[] theDS = new DataSource[2];  
theDS[0] =  
    Manager.createDataSource(audioCDI.getLocator());  
theDS[1] =  
    Manager.createDataSource(videoCDI.getLocator());
```

- Build a merge DataSource:

```
DataSource ds =  
    Manager.createMergingDataSource(theDS);
```

First problem solved

- To display the captured video, find a `Player` and associate a `ControllerListener` by:

```
Player dualPlayer =  
    Manager.createPlayer(ds);  
dualPlayer.addControllerListener(...);  
dualPlayer.start();
```

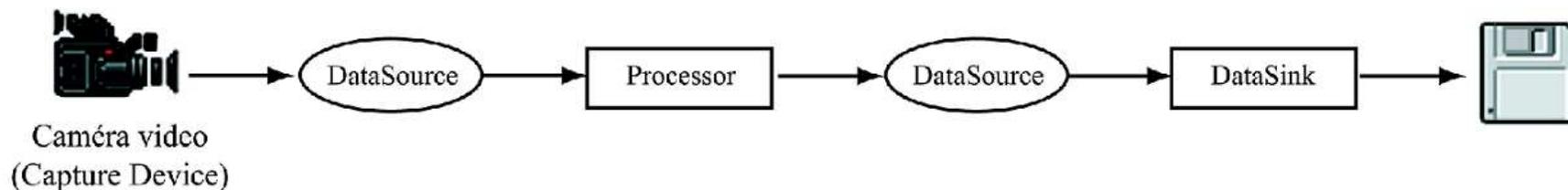
Second problem solved

- The demo
- Please smile ;-)

9 lanceCapture.bat
seconde partie

Save a video: the two main characters

- You must have an object which takes the `DataSource` and build a multimedia format: this object is a `Processor`
- We need an object which takes a `DataSource` and put it in a given place: this object is an instance of a class which implements `DataSink`



Save a video: two other actors

- The location of media content is abstracted by a `MediaLocator`
- A `FileTypeDescriptor` abstracts the format to create:

```
FileTypeDescriptor ftd = new  
    FileTypeDescriptor(FileTypeDescriptor.QUICKTIME);
```

for MOV (Quicktime) format

- or

```
FileTypeDescriptor.MSVIDEO
```

for AVI format

Code to save a video

```
// The DataSource ds is the source of multimedia datas filled
// by the camera
// We are looking for a Processor which can build a MOV DataSource
String outputFormat = FileTypeDescriptor.QUICKTIME;
FileTypeDescriptor ftd = new FileTypeDescriptor(outputFormat);
Format[] formats = new Format[]{new AudioFormat(null),
                               new VideoFormat(null)};
ProcessorModel pm = new ProcessorModel(ds, formats, ftd);
Processor processor = Manager.createRealizedProcessor(pm);

// Now we take the output of the processor and we put in a file
DataSource outputDS = processor.getDataOutput();
MediaLocator ml = new MediaLocator("file:" + FILE_NAME + ".mov");
DataSink datasink = Manager.createDataSink(outputDS, ml);

// We launch DataSink and the Processor
datasink.open();
datasink.start();
processor.start();
```

Stop the saving

```
if (processor != null) {  
    processor.stop();  
    processor.close();  
}  
if (datasink != null)  
    datasink.close();  
  
processor = null;
```

Bibliographie

- <http://java.sun.com/products/java-media/jmf/index.html> : les technologies Java pour les données temporelles
- <http://java.sun.com/javase/technologies/desktop/media/jmf/reference/faqs/index.html> une bonne FAQ concernant les données temporelles.
- La documentation des APIs JMF :
<http://java.sun.com/products/java-media/jmf/2.1.1/apidocs/overview-summary.html>

Bibliographie (suite)

- <http://java.sun.com/javase/technologies/desktop/media/jmf/1.0/guide/index.html>
: le guide de programmation Java Media Framework
- Programming with the Java Media Framework ; Sean C. Sullivan, Loren Winzeler, Jeannie Deagen, Deanna Brown, ed Wiley. ISBN 0-471-25169-0

Fin