

# Introduction à Java Media Framework et Java 1.2

**[www.Mcours.com](http://www.Mcours.com)**  
Site N°1 des Cours et Exercices Email: [contact@mcours.com](mailto:contact@mcours.com)

# Présentation

Java Media Framework (JMF) est une architecture unifiée pour la synchronisation, le traitement, l'affichage de données temporelles comme les données audio, la vidéo, le format MIDI, etc. à l'intérieur d'applications indépendantes ou d'applets.

utilise Java 1.1 et suivants

développé par Sun, Silicon Graphics et Intel.

constitué de 3 étapes : Player, Capture, Conference.

Java Media Player est la première étape et comporte la synchronisation, l'exécution, la présentation et le stockage de données temporelles compressées.

L'URL de référence est :

<http://java.sun.com/products/java-media/jmf/>

elle indique les spécifications de JMF 1.0.

Le soft et les exemples pour Solaris et Windows peuvent être chargés à partir de <http://developer.javasoft.com/developer/earlyAccess/jmf.html> (souscrire à Java Developer Corner mais c'est gratuit)

# Java Media Player

supporte les formats :

pour l'audio: AIFF, AU, DVI, G.723, GSM, IMA4, MIDI, MPEG-1 Layer 1/2, PCM, RMF, WAV

pour la vidéo: Apple Graphics (SMC), Apple Animation (RLE) Cinepak, H.261, H.263, Indeo 3.2, Motion-JPEG, MPEG-1, Uncompressed

les formats de fichiers : AVI, QuickTime, Vivo

les protocoles File, FTP, HTTP, RTP (RFC 1889/1890)

n'inclus pas la capture de données qui doit être la partie de Java Media Capture. Par contre on peut capturer des sons à l'aide Java Sound voir à

<http://java.sun.com/products/java-media/sound/techReference/javasoundfaq.html>

# Java 1.2

en version beta 3 le 25 Mars 1998.  
disponible pour Win32 (15 Mo), solaris2-  
sparc et x86 à partir du Java Developer  
Connection (s'inscrire et c'est gratuit) à :  
<http://developer.javasoft.com/developer/earlyAccess/jdk12/index.html>

La documentation est accessible directement  
à :  
<http://www.javasoft.com/products/jdk/1.2/docs/index.html>



## Java 1.2 (suite)

Propose :

- 1.Security Enhancements
- 2.Swing (JFC)
- 3.Java 2D (JFC)
- 4.Accessibility (JFC)
- 5.Drag and Drop (JFC)
- 6.Collections
- 7.Java Extensions Framework
- 8.JavaBeans Enhancements
- 9.Input Method Framework
- 10.Package Version Identification
- 11.RMI Enhancements
- 12.Serialization Enhancements
- 13.Reference Objects
- 14.Audio Enhancements
- 15.Java IDL
- 16.Performance Enhancements
- 17.JAR Enhancements
- 18.JNI Enhancements
- 19.Reflection Enhancements
- 20.JVMDI: A New Debugger Interface
- 21.JDBC Enhancements

### 1.Security Enhancements.

À tout code chargé, la politique de sécurité locale lui affecte des permissions pour les ressources (lire, écrire dans un fichier ou répertoire, se connecter sur un hôte distant sur l'un de ses ports, ...). La politique de sécurité locale indique qu'elles sont les permissions pour des entités signataires.

## Audio Enhancements

support pour l'audio dans les applications comme dans les applets.

permet d'utiliser les formats audio :

MIDI (type 0 et type 1), RMF, WAV, AIFF, and AU

intègre l'API Java Sound qui continue à se développer.

voir à

<http://www.javasoft.com/products/java-media/sound/index.html>

## Swing (JFC)

JFC = Java Foundation Classes

= ensemble de services et de composants

graphiques en Java : cf. 2, 3, 4, 5 ci dessus.

cf. FAQ à

<http://java.sun.com/products/jfc/faq.html>

l'ensemble des composants graphiques est Swing. Ces composants ont le "look and feel" de la plateforme qui exécute le code Java.

Swing est un sous ensemble du projet JFC.

Swing = les composants graphiques de JFC.

JFC étend les fonctionnalités de AWT (composants légers, ...).

IFC de netscape (aussi 100% pure Java) vont migrer vers JFC (accord Sun-netscape).

## Java 2D (JFC)

ajoute de nouvelles fonctionnalités graphiques (manipulation de la couleur, transformations géométriques, ...) à `java.awt` pour les dessins, le texte, les images, etc. pour les applications comme pour les applets.

complètement compatible avec Java 1.1.

utilise la classe `Graphics2D` pour indiquer :

- l'épaisseur des lignes
- le type de jointure entre lignes
- le remplissage par couleur ou texture
- l'anti-aliasing
- ...

En Java 1.1 on écrit :

```
public void paint(Graphics g) {  
    g.setColor(Color.red);  
    g.fillRect(300, 300, 200, 100);  
}
```

qui dessine un rectangle rouge.

## Graphics2D

En Java 1.2, on peut préciser le style de la couleur de remplissage, définir des polygones complexes, etc. grâce à un objet de classe `java.awt.Graphics2D` (sous classe de `java.awt.Graphics`) par un changement de type :

```
public void paint(Graphics g) {  
    Graphics2D g2d = (Graphics2D) g;  
  
    g2d.setColor(Color.red);  
  
    GeneralPath path = new  
    GeneralPath(GeneralPath.EVEN_ODD);  
    path.moveTo(300.0f, 400.0f); // coin bas gauche  
    path.lineTo(500.0f, 400.0f); // ligne vers la droite  
    path.lineTo(500.0f, 300.0f); // ligne vers le haut  
    path.lineTo(300.0f, 300.0f); // ligne vers la gauche  
    path.closePath(); // on referme le trace  
  
    g2d.fill(path); // on remplit en rouge.  
}
```

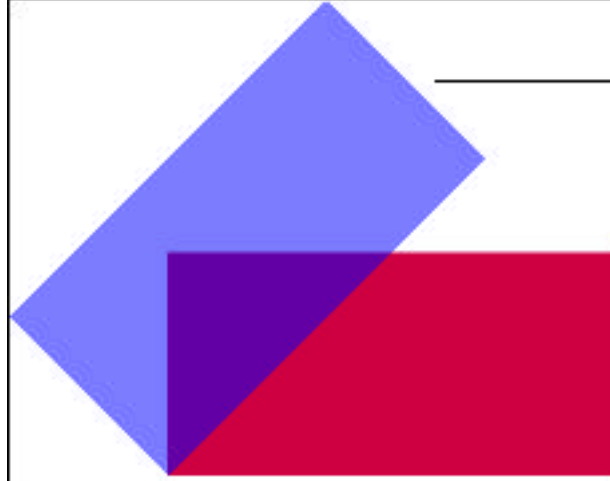
on dessine en fait un chemin (objet de la classe `GeneralPath`) et après l'avoir refermé, on le remplit. Le mode de remplissage est indiqué dans le constructeur du chemin.

On peut évidemment dessiner des rectangles qui sont maintenant des objets de classe qui implémentent l'interface `Shape`.



# manipulations avancées

transformations affines (translation, rotation), transparence :



```

public void paint(Graphics g) {
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.red);
    GeneralPath path = new
    GeneralPath(GeneralPath.EVEN_ODD);
    path.moveTo(0.0f, 0.0f); // lower left corner
    path.lineTo(200.0f, 0.0f); // lower right corner
    path.lineTo(200.0f, -100.0f); // upper right corner
    path.lineTo(0.0f, -100.0f); // upper left corner
    path.closePath(); // close the rectangle
    AffineTransform at = new AffineTransform();
    at.setToTranslation(300.0, 400.0);
    g2d.transform(at);
    g2d.fill(path);

    // second rectangle
    g2d.setColor(Color.blue); // il sera bleu
    // positionne la composante alpha de transparence a
    // 50% de transparence
    AlphaComposite comp =
    AlphaComposite.getInstance(AlphaComposite.SRC_OVER,
    0.5f);
    g2d.setComposite(comp);
    // rotation de -PI/4 radians (i.e. -45°).
    // L'orientation est celle des aiguilles d'une montre !!
    at.setToRotation(-Math.PI/4.0);
    g2d.transform(at);
    // affichage du rectangle bleu
    g2d.fill(path);
}

```

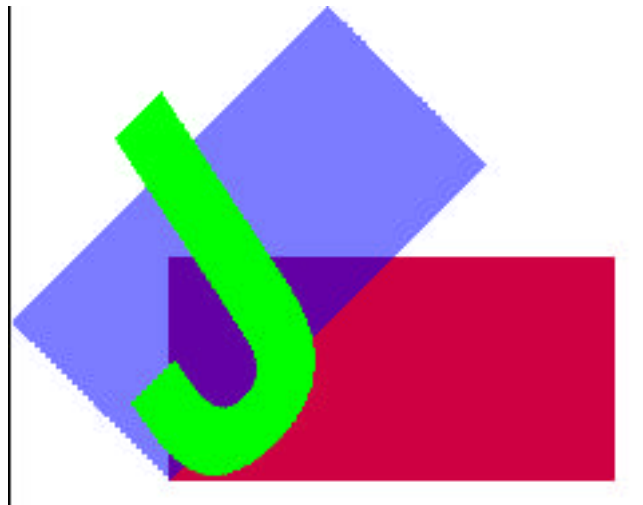
## Le texte

Le texte est traité comme un dessin. Par exemple pour dessiner la lettre J en police Helvetica grasse oblique, après une rotation de 45° "à gauche" sur les rectangles précédent, il suffit d'ajouter :

```
// construire l'objet police Helvetica-BoldOblique de 200 points
Font myFont = new Font("Helvetica-BoldOblique",
                        Font.PLAIN, 200);

// afficher le caractere `J' en vert, opaque
// apres rotation de 45°
// a gauche en utilisant le gc precedent.

g2d.setColor(Color.green);
g2d.setComposite(AlphaComposite.getInstance(AlphaComposite.SRC_OVER, 1.0f));
g2d.drawString("J", 0f, 20f);
```



# Les images

De même que pour les dessins et les figures, on peut appliquer des transformations affines sur des images et écrire :

```
Image image = applet.getImage(url);
AffineTransform at = new AffineTransform();
at.rotate(Math.PI/4.0);
g2d.drawImage(image, at, this);
```

Les pixels des images peuvent avoir une valeur de transparence distincts 2 à 2 comme :



# **Bibliographie**

**Java 1.2 :**

<http://java.sun.com/products/jdk/1.2/docs/index.html>

**pour Java 2D :**

<http://java.sun.com/products/jdk/1.2/docs/guide/2d/spec/j2d-title.fm.html>

## **Java, les animations et le multimédia**

On trouve sur Internet certains sites proposant des programmes Java (souvent des applets) qui intègrent, le mouvement, le changement d'images, parfois les sons, etc. Malheureusement très souvent ces pages demande beaucoup de temps de chargement.

<http://www.worldnet.net/~tomsoft/Java/View3D/View3D.html> propose des objets 3D (avion, hélicoptère, dirigeable, etc. ) à manipuler avec la souris. propose des effets d'ombres, des vues fils de fer, etc.

<http://www.developer.com/directories/pages/dir.java.mm.anim.video.html>  
un catalogue remis à jour quotidiennement des meilleures applets d'animation et multimédia.

<http://www.boutell.com/baklava/scones.html>

le célèbre jeu "casse bouteille" spatial.

# Bibliographie

## Java, les animations et le multimédia (suite)

<http://junior.apk.net/~jbartan/idiot/idiot.html>

un bon gag : un bouton poussoir qui fuit quand on essaie de cliquer dessus (mais parfois on y arrive !).

<http://www.best.com/~pvdl/rubik.html>

le cube de Rubik en Java : magnifique !

plus généralement le site de Karl Hörnell à

<http://www.tdb.uu.se/~karl/> est très beau

avec de magnifique applets à

<http://www.tdb.uu.se/~karl/brain.html>

entre autre un éditeur de partition et

exécuteur de mélodies à :

<http://www.tdb.uu.se/~karl/java/bluenotes.html>

