CHAPITRE : Introduction aux applets

Une Applet = **Appl**ication Intern**et** écrite en Java intégrée à une page HTML ne pouvant être qu'exécutée par un navigateur et s'affichant dans la page HTML.

Une Applet est un programme Java (sans méthode main) exécuté dans une page Web ou par un programme **applet viewer.**

Une page HTML est un fichier texte contenant des descriptions de la mise en page du texte et des images. Ces descriptions sont destinées au navigateur afin qu'il assure l'affichage de la page, elles s'effectuent par l'intermédiaire de balises (parenthèses de description contextuelles). Un fichier HTML commence toujours par la balise <HTML> et termine par la balise </HTML>.

Une page HTML minimale contient deux sections :

- l'en-tête de la page balisée par <HEAD> ...</HEAD>
- le corps de la page balisé par <BODY> ... </BODY>

Voici un fichier texte minimal pour une page HTML :

```
<HTML>
    <HEAD>
    </HEAD>
    <BODY>
    </BODY>
</HTML>
```

Une applet Java est invoquée grâce à deux balises spéciales insérées au sein du corps de la page HTML : <APPLET CODE =.....> </APPLET>

6.1. La classe java.applet.Applet

Voici la hiérarchie des classes dont Applet dérive :

```
java.lang.Object
  |
  +--java.awt.Component
       |
```

```
+--java.awt.Container
    |
    +--java.awt.Panel
        |
        +--java.applet.Applet
```

Minimum requis pour une applet :

Si vous voulez écrire des applets, il faut posséder un navigateur pour exécuter les applets et un éditeur de texte permettant d'écrire le code source Java.

Vous pouvez aussi utiliser un environnement de développement Java

Tout environnement de développement java doit contenir un moyen de visionner le résultat de la programmation de votre applet, cet environnement fera appel à une visionneuse d'applet (dénommée AppletViewer dans le JDK).

Votre applet doit hériter de la classe Applet

public class AppletExemple **extends** Applet { .....
}
Comme toutes les classes Java exécutables, le nom du fichier doit correspondre très exactement au nom de la classe avec comme suffixe java (ici : **AppletExemple.java**)

6.2. Fonctionnement d'une applet

6.*2.1 La méthode init( )*

La méthode **init( )** est appelée **une seule fois**, lors du chargement de l'applet avant que celui-ci ne soit affiché. Lorsqu'une applet s'exécute la méthode principale qui s'appelait **main( )** dans le cas d'une application Java se dénomme ici **init( )**. Il nous appartient donc de **surcharger dynamiquement (redéfinir)** la méthode **init** de la classe Applet afin que notre applet fonctionne selon nos attentes.

Exemple d'applet vide :

```
import java.applet.*;
public class AppletExemple1 extends Applet
{
}
```

6.2.2 La méthode start( )

La méthode **start( )** est appelée **à chaque fois**, soit :

- après la méthode init
- après chaque modification de la taille de l'applet (minimisation,...).

6.2.3 La méthode paint( )

La méthode **paint(Graphics x)** est appelée **à chaque fois**, soit :

- après que l'applet a été masquée, déplacée, retaillée,...
- à chaque réaffichage de l'applet (après minimisation,...).

6.2.4 Les méthodes stop( ) et destroy( )

La méthode **stop( )** est appelée **à chaque fois**, soit :

- que l'applet a été masquée dans la page du navigateur (défilement vertical ou horizontal dans le navigateur), déplacée, retaillée,...
- lors de l'abandon et du changement de la page dans le navigateur

La méthode **destroy( )** est appelée **à chaque fois**, soit :

- que l'utilisateur charge une nouvelle page dans le navigateur
- lors de la fermeture du navigateur

6.3. Une applet dans une page HTML

6.*3.1 Le code d'appel d'une applet*

Le code d'appel d'une applet est intégré au texte source de la page dans laquelle l'applet va être affichée. Ce sont les balises <APPLET CODE =.....> et </APPLET> qui précisent les modalités de fonctionnement de l'applet. En outre, l'applet s'affichera dans la page exactement à l'emplacement de la balise dans le code HTML. Donc si l'on veut déplacer la position d'une applet dans une page HTML, il suffit de déplacer le code compris entre les deux balises <APPLET CODE =.....> et </APPLET>.

Voici une page HTML dont le titre est "Exemple Applet " et ne contenant qu'une applet

```
<HTML>
  <HEAD>
        <TITLE> Exemple Applet </TITLE>
  </HEAD>

  <BODY>
    <APPLET CODE="NomApplet.class" WIDTH=200 HEIGHT=100>
    </APPLET>
  </BODY>

          </HTML>
```

Il y a donc des paramètres obligatoires à transmettre à une applet, ce sont les paramètres CODE, WIDTH et HEIGHT :
CODE : le nom du fichier contenant la classe applet à afficher.

WIDTH : la largeur de l'applet au démarrage dans la page HTML (en pixels)

HEIGHT : la largeur de l'applet au démarrage dans la page

6.4. Quelques exemples sur les applets :

6.4.1. Premier exemple : quelques figures simples

```
/* Premier exemple simple sur Applet :

  1) Applet : petite application exécutée par un navigateur Web
          (exemple Netscape, Explorer, ....)
          La méthode principale "void main" ne se présente pas.

  2) AWT : Abstract Windowing Toolkit : un ensemble de classes
       pour créer, surtout, une interface utilisateur graphique
        Graphics est une classe de AWT

  3) classe Applet : classe mère
        Les applets sont des sous-classes de la classe Applet
        du paquetage (package) "java.applet"

  C'est la raison des deux lignes d'importation
```

Syntaxe de quelques méthodes appliquées sur un objet (exemple g)
de la classe "Graphics" :
(dessiner) afficher une chaîne à partir d'une position
g.drawString(une chaîne de caractères, abscisse, ordonnée);

tracer une ligne reliée 2 points A et B
g.drawLine (A.x,A.y, B.x, B.y);

dessiner un rectangle ABCD :

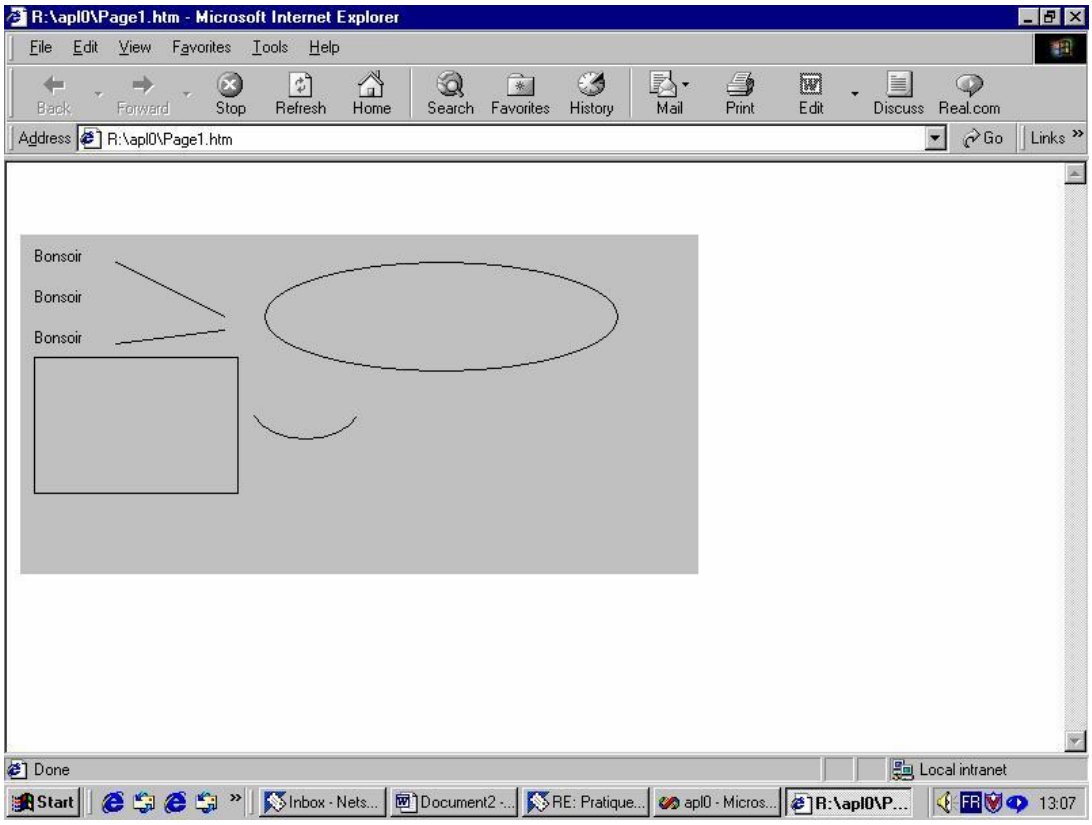
A **B**

D **C**

g.drawRect(A.x, A.y, largeur AB,hauteur BC);

dessiner un "arc" (d'une ellipse à l'intérieur d'un rectangle)
à faire : changer des paramètres, exécuter, observer pour comprendre
le rôle de ces paramètres
g.drawArc(A.x,A.y,largeur rectangle,hauteur rectangle, angle0, angle ouverture);
*/

```
import java.awt.*;
import java.applet.*;

public class Applet1 extends Applet
{
        public void paint(Graphics g) {

           g.drawString("Bonsoir", 10, 20);
           g.drawString("Bonsoir", 10, 50);
           g.drawString("Bonsoir", 10, 80);

           g.drawLine (70,20, 150, 60);
           g.drawLine (70,80, 150, 70);

           g.drawRect(10, 90, 150,100);

           g.drawArc(170,100,80,50, 200, 140);
           g.drawOval(180,20, 260, 80); }}
```

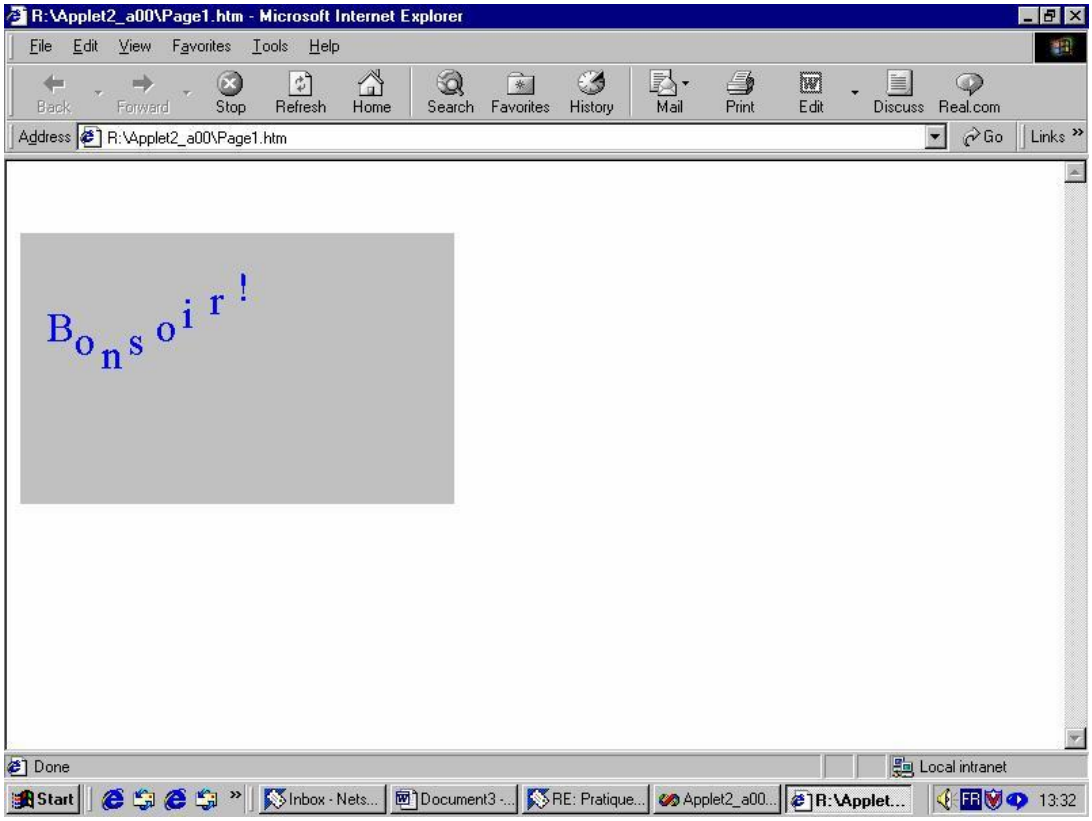


Voici le fichier HTML qui lance cette applet :

```
<HTML>
        <HEAD>
           <TITLE> Exemple Applet </TITLE>
        </HEAD>
        <BODY>
            <applet code= “Applet1.class” width=320 height=200 >
                <param name=background value="008080">
                <param name=foreground value="FFFFFF">
            </applet>
        </BODY>
</HTML>
```

6.4.2) Deuxième exemple : quelques figures simples

```
/* Deuxième exemple simple sur Applet : affichage "ondulé" d'une chaîne de
caractères
*/
import java.awt.*;
import java.applet.*;
public class Applet2 extends Applet
{
   //Phase initialisation d'une applet quand elle est chargée
        public void init() {

                /* Le constructeur Font de la classe Font à 3 paramètres :
                  Font (police, style, taille)
                Quelques exemples sur style :
                Font.ITALIC   : italique
                Font.BOLD     : en gras
                Font.PLAIN      : normal

                Font.BOLD + Font.ITALIC   : italique en gras
                etc ....
                */
                 setFont(new Font("TimesRoman", Font.PLAIN, 30));
}
        public void paint(Graphics g) {

                // chaîne de caractères à afficher
                String souhait = "Bonsoir!" ; // 8 caractères

        /* position de chaque lettre de la chaîne : tableau
                        à deux indices :
              */
                int [][] position = { {20, 80 },{40, 90 },{ 60, 100 },
                        {80, 90 },{100, 80 },{ 120, 70 },
                                                {140,60 },{160,50 }};
                // imposer la couleur (ici bleue)
                g.setColor(Color.blue);

                // affichage avec la méthode drawString
                for (int i = 0 ; i < souhait.length() ; i++)
                        g.drawString( souhait.substring(i, i+1),
                              position[i][0],   position[i][1]); }}
```



6.4.3) Troisième exemple : dessin d'un sourire!

```
/* Troisième exemple simple sur Applet : un sourire
*/
import java.awt.*;
import java.applet.*;

public class Applet3 extends Applet
{       static void dessinerSourire(Graphics g) {

                // dessiner la figure
                g.drawOval(50, 40, 100, 120);

                // coin en haut, à gauche (pour l'oeil gauche)
                // de la classe Point : construire un point
                Point p = new Point(70, 80);
```

```
            /* notez que les 2 champs de données x et y de la classe
               Point ont l'accès "public" qui est un cas très spécial
               g.fillRect(point.x, point.y, longueur, largeur);
               ici on  dessine l'oeil gauche
            */
            g.fillRect(p.x, p.y, 10, 10);

            // faire une translation pour le coin gauche, en haut
            // de l'oeil droit
            p.translate(50,0);
            // dessiner l'oeil droit
            g.fillRect(p.x, p.y, 10, 10);

            // dessiner le sourire (rire à toutes les dents? (2 derniers
            // paramètres
            g.drawArc(60,50, 70, 90, 200, 140);

    }

    // un cas de polymorphisme : surdéfinition de méthode
    static void afficher(Graphics g, String texte, int x, int y) {
    g.setFont(new Font("Courier New", Font.ITALIC+Font.BOLD, 25));
        g.setColor(Color.blue);
        g.drawString(texte, x, y);
    }

    // un cas de polymorphisme : surdéfinition de méthode
    static void afficher(String texte, Graphics g, int x, int y) {
        g.setColor(Color.black);
            g.setFont(new Font("Courier New", Font.PLAIN, 14));

            // à essayer de comprendre en observant l'exécution
        for (int i = 0 ; i < texte.length() ; i++)
            g.drawString(texte.substring(i, i+1), (x += 10),
                            ((i >=texte.length()/2) ? (y +=10):(y -=10)));

    }
```

```
public void paint(Graphics g) {

                dessinerSourire(g);
              g.setColor(Color.darkGray);
              g.drawLine(180,    0, 180, 400);

              // déjà vue
              afficher(g, "Vitamine S : un sourire!", 200, 50);

              // un affichage spécial
              afficher("Droit d'auteur à débarasser!", g, 200, 250);
    }
}
```