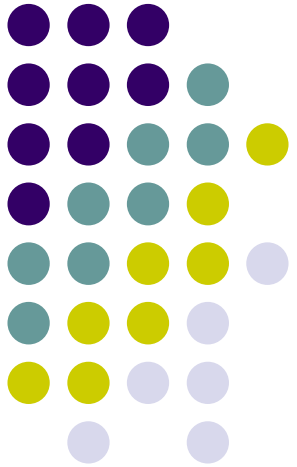
第二十一章
網頁的精靈-applet

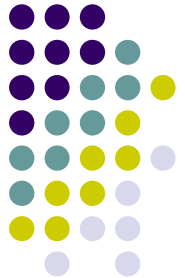


applet概述

認識applet的執行程序

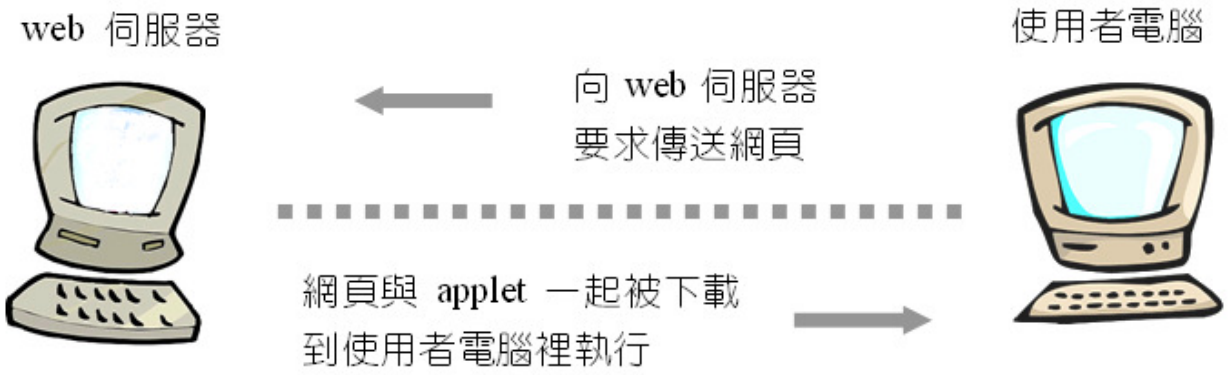
學習載入影像與簡單的動畫製作

學習在applet裡播放音樂檔



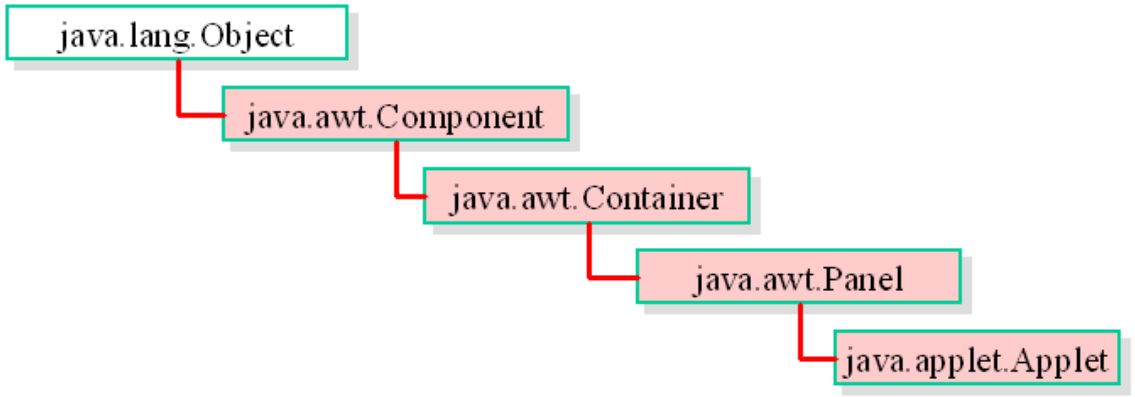
認識applet

- applet執行的流程
  - applet撰寫、編譯
  - 當使用者連到這個網頁裡
  - applet便會隨著網頁下載
  - 再於使用者的電腦執行，過程如下：



Applet類別的繼承關係

- java.applet.Applet類別
  - 用來處理applet的運作
  - Applet類別繼承自Panel類別，其繼承關係如下圖：



簡單的applet (1/2)

App21_1.java是個簡單的applet程式

```
01  // App21_1, 簡單的 applet 程式
02  import java.awt.*;
03  import java.applet.Applet;                  // 載入 Applet 類別
04
05  public class App21_1 extends Applet         // App21_1 衍生自 Applet 類別
06  {
07     public void paint(Graphics g)
08     {
09        g.setColor(Color.blue);               // 設定繪圖顏色為藍色
10        g.fillOval(30,30,50,50);              // 繪出圓形並填滿藍色
11        g.setColor(Color.orange);             // 設定繪圖顏色為橘色
12        g.fillOval(60,40,90,90);              // 繪出圓形並填滿橘色
13     }
14  }
```

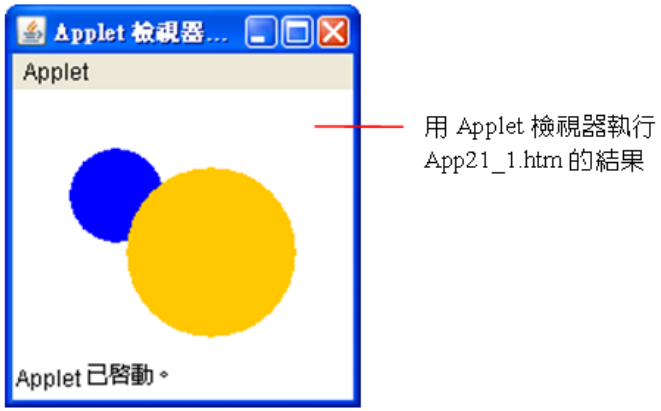
- 撰寫與編譯App21_1.java後，還須撰寫一個HTML檔把applet嵌進入

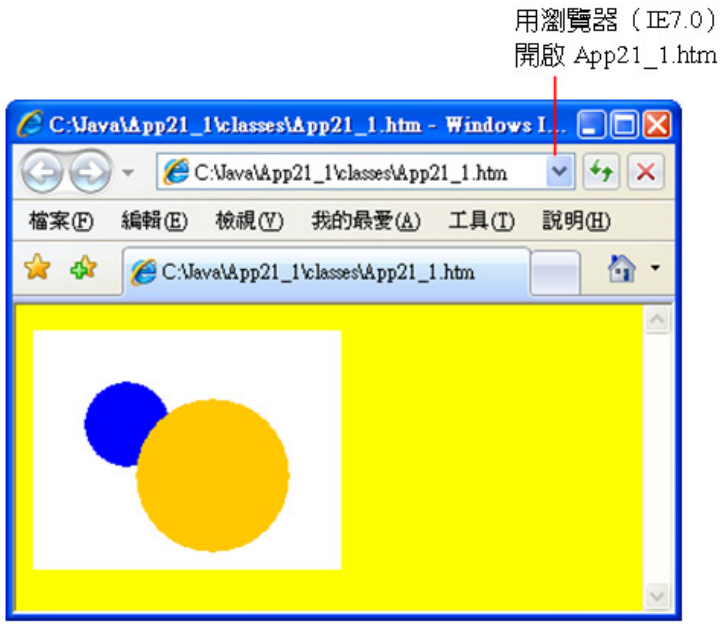
```
01  <!-- App21_1.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >        —— 設定背景顏色
04  <APPLET
05     CODE     = "App21_1.class"     —— 指定 applet 為 App21_1.class
06     WIDTH    = "180"               } 設定 applet 的寬度為 180 個像素，
07     HEIGHT   = "140"  >            } 高度為 140 個像素
08  </APPLET>
09  </BODY>
10  </HTML>
```

App21_1.htm列出HTML檔的內容

簡單的applet (2/2)

- 觀看applet的執行結果
  - 可以用Applet檢視器
  - 或是由瀏覽器開啟App21_1.htm來查看





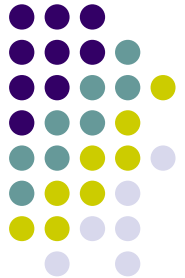
- paint() method被呼叫的時機：
  - 從縮小圖示還原之後
  - 新建的視窗顯示於螢幕上、或從隱藏變成顯示時
  - 正在改變視窗大小時

Applet類別的method (1/2)

- 撰寫applet程式時必須做下列幾件事：
  - 載入java.applet.Applet類別
  - 定義一類別繼承自Applet類別
  - 把相關的程式碼撰寫在這個類別內
- 下表列出Applet類別常用的method：

表 21.1.1　java.applet.Applet 常用的 method

method	主要功能
void destroy()	銷毀 Applet 元件，通常在呼叫此 method 之前，會先呼叫 stop()。
String getAppletInfo()	取得 Applet 元件的資訊，如作者、版權與版本等。如果沒有這些資訊，則傳回 null
AudioClip getAudioClip(URL url)	取得網址 url 上的 AudioClip 物件(聲音檔)
AudioClip getAudioClip(URL url, String str)	取得網址為 url，名稱為 str 的聲音檔
URL getCodeBase()	取得此 applet 所在之網址（路徑）
URL getDocumentBase()	取得嵌入此 applet 之文件的網址（路徑）



Applet類別的method (2/2)

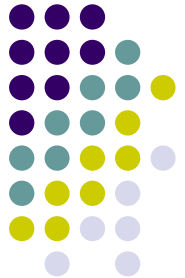
Image getImage(URL url)	取得網址為 url 的影像檔
Image getImage(URL url, String name)	取得網址為 url，名稱為 name 的影像檔
String getParameter(String name)	取得名稱為 name 的引數
String[ ][ ] getParameterInfo()	取得引數訊息，並以字串陣列的方式傳回
void init()	瀏覽器載入 applet 時，所呼叫的初始化 method
boolean isActive()	測試 applet 是否在正在執行
static AudioClip newAudioClip(URL url)	取得網址為 url 的聲音檔
void play(URL url)	播放網址為 url 的聲音檔
void play(URL url, String name)	播放網址為 url，名稱為 name 的聲音檔
void resize(int width, int height)	改變 applet 的大小，高為 height，寬為 width
void showStatus(String msg)	顯示訊息 msg 在狀態視窗上
void start()	當 applet 處於作用中視窗時會執行此 method
void stop()	當 applet 所在的網頁被切換到別的網頁，或最小化時所呼叫的 method

HTML的檔案結構

- HTML文件
  - 屬於純文字檔
  - HTML是由許多標記（tags）所組成，標記必須成對
  - 內容是用來控制網頁要如何呈現
  - 下面的格式說明HTML 檔案的基本架構：

HTML 網頁的基本架構

```
<HTML>
<BODY BGCOLOR = "顏色值" >
<APPLET
    <!--此處撰寫applet的內容-->
>
</APPLET>
</BODY>
</HTML>
```



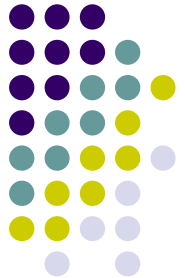
撰寫HTML的注意事項

- 撰寫HTML文件時，必須注意下面幾個事項

(1) HTML的註解文字是以「<!--」符號為開頭，以「-->」符號為結尾

(2) HTML檔案並無大小寫之分

(3) 每一個HTML的標記必須放在「< >」符號裡，而結尾標記必須加上「/」符號，如 </HTML>



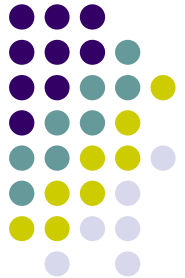
HTML的<APPLET>標記 (1/2)

- <APPLET> 標記的格式如下：

<APPLET>標記的格式

```
<APPLET
    CODE = "filename"
    WIDTH = "pixels"
    HEIGHT = "pixels"
    [CODEBASE = "URL"]
    [ALT = "alternate text"]
    [NAME = "instance name"]
    [ALIGN = "alignment"]
    [VSPACE = "pixels"]
    [HSPACE = "pixels"]
>
    [<PARAM NAME = "name" VALUE = "value">]
       ....
    [<PARAM NAME = "name" VALUE = "value">]
</APPLET>
```

HTML的<APPLET>標記 (2/2)



- 下表是<APPLET>標記裡每一個參數所代表的意義：

表 21.1.2 <APPLET>標記之參數說明

參數	主要功能
CODEBASE	指定此 applet 所在之網址（路徑），若未設定此項，則以目前執行的目錄為 applet 的路徑
CODE	設定要開啟之 applet 的檔案名稱，注意必須包含副檔名.class
ALT	如果瀏覽器無法顯示 applet，則以 alternateText 字串來顯示
NAME	設定 applet 的名稱。每一個 applet 如果都有一個名稱，則兩個以上的 applet 要相互參考時便可利用名稱來指定
WIDTH	設定 applet 顯示的寬度，單位為像素
HEIGHT	設定 applet 顯示的高度，單位為像素
ALIGN	設定對齊方式，包括 LEFT、RIGHT、TOP、BOTTOM 與 MIDDLE
VSPACE	設定 applet 上下所保留的寬度
HSPACE	設定 applet 左右所保留的寬度
PARAM NAME	要傳給 applet 的參數名稱
VALUE	要傳給 applet 的參數值

簡單的範例 (1/2)

- 再介紹一個簡單的範例

```
01  <!-- App21_2.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05      CODE     = "App21_2.class"
06      WIDTH    = "180"
07      HEIGHT   = "120"
08      ALT      = "很抱歉，您的瀏覽器不支援 Java applet"
09      ALIGN    = "RIGHT"
10      VSPACE   = "20" >
11  </APPLET>
12  </BODY>
13  </HTML>
```

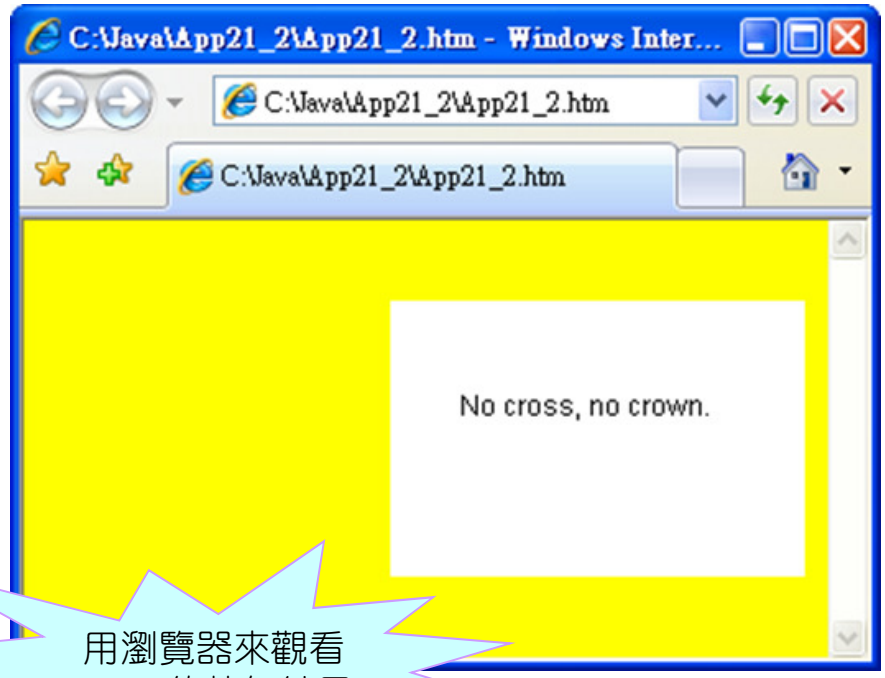
設定applet靠右對齊

設定applet與其它元件的垂直距離為20

```
01  // App21_2, 簡單的 applet 程式
02  import java.awt.*;
03  import java.applet.Applet;
04
05  public class App21_2 extends Applet
06  {
07     public void paint(Graphics g)
08     {
09        g.drawString("No cross, no crown.",30,50); // 在繪圖區內寫上字串
10     }
11  }
```

簡單的範例 (2/2)

- 分別用瀏覽器及Applet檢視器觀看執行結果



傳遞參數到applet

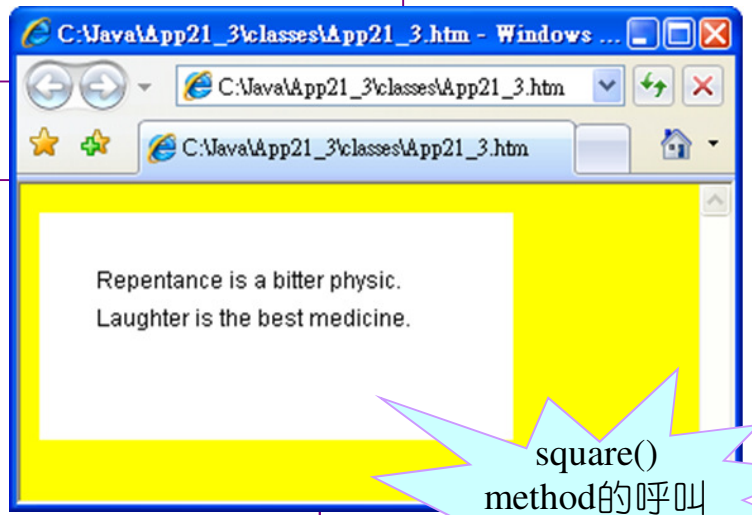
App21_3是參數傳遞的範例

```
01  // App21_3, 簡單的 applet 程式
02  import java.awt.*;
03  import java.applet.Applet;
04
05  public class App21_3 extends Applet
06  {
07     public void paint(Graphics g)
08     {
09        g.drawString(getParameter("str1"),30,40); // 取得 HTML 裡的 str1 字串
10        g.drawString(getParameter("str2"),30,60); // 取得 HTML 裡的 str2 字串
11     }
12  }
```

參數傳到applet裡

- ✓ 利用<PARAM>標記
- ✓ getParameter() method接收參

```
01  <!-- App21_3.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05     CODE     = "App21_3.class"
06     WIDTH    = "250"
07     HEIGHT   = "120" >
08
09  <PARAM NAME= "str1" VALUE = "Repentance is a bitter physic." >
10  <PARAM NAME= "str2" VALUE = "Laughter is the best medicine." >
11  </APPLET>
12  </BODY>
13  </HTML>
```



square() method的呼叫過程

使用Java主控台

- Java applet可利用System.out.println() method，把字串輸出到「主控台」(console)
- 開啟「Java控制面版」對話方塊

Windows的「控制台」裡有一個「Java」的圖示，執行它之後可開啟「Java控制面版」對話方塊

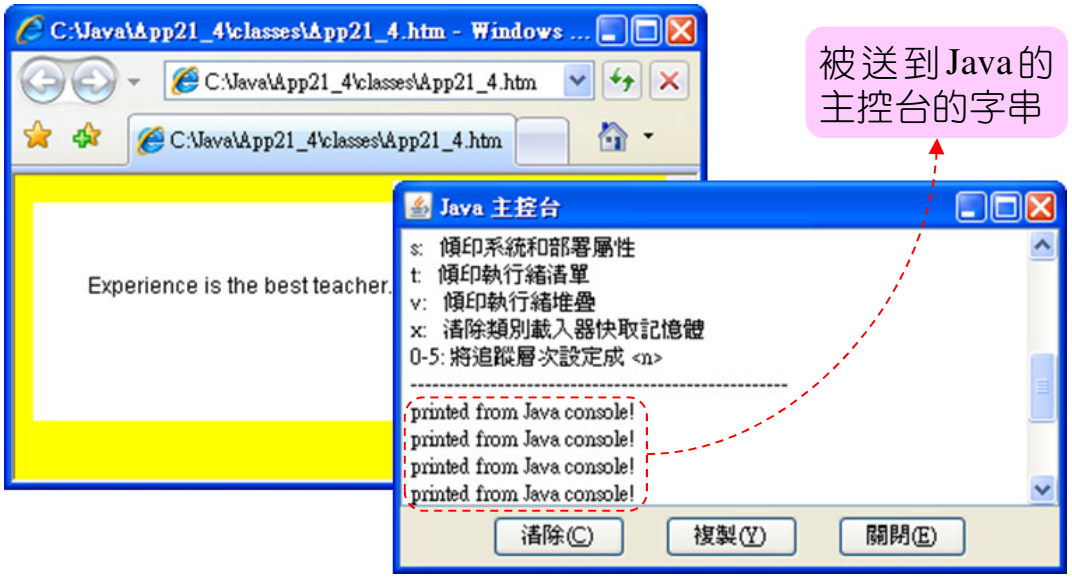
Java主控台的實例

- App21_4是Java主控台的使用範例

```
01  // App21_4, 簡單的 applet 程式
02  import java.awt.*;
03  import java.applet.Applet;
04
05  public class App21_4 extends Applet
06  {
07     public void paint(Graphics g)
08     {
09        g.drawString("Experience is the best teacher.",30,50);
10        System.out.println("printed from Java console!");
11     }
12  }
```

到控制台的「Java」-「進階」裡設定「顯示主控台」，Java的主控台才會顯示

將字串送到 Java 的主控台



被送到Java的主控台的字串

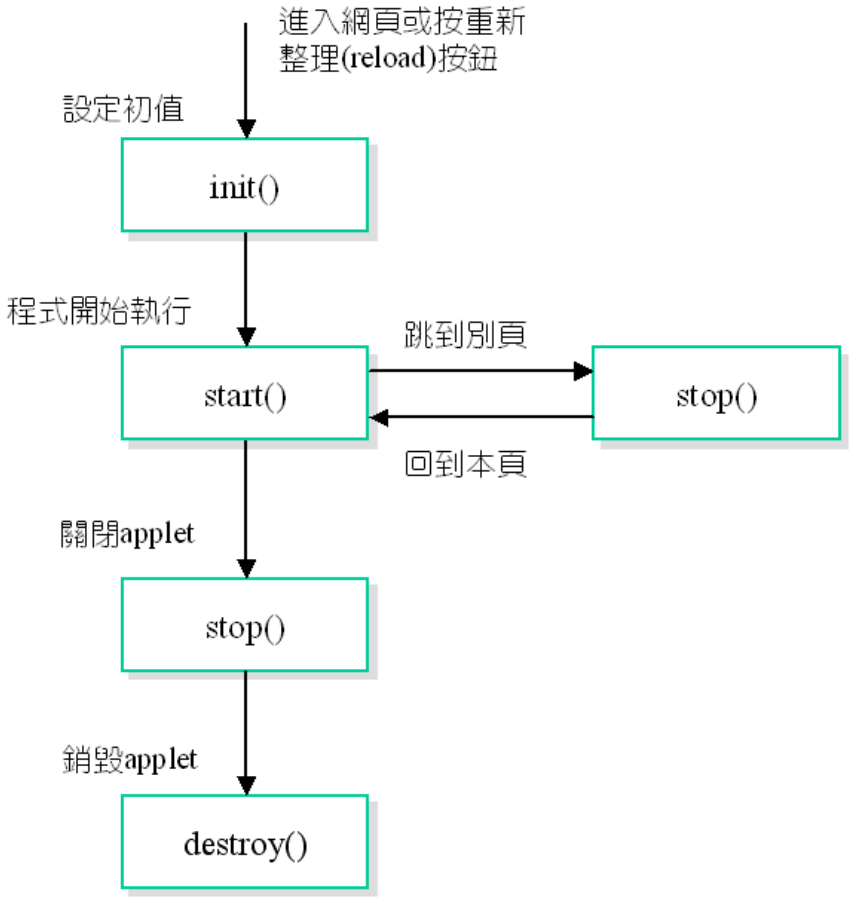
```
01  <!-- App21_4.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05     CODE     = "App21_4.class"
06     WIDTH    = "250"
07     HEIGHT   = "120"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```

applet的執行程序

- Applet類別裡有init()、start()、stop()與destroy() method
- 下面是這四個method被呼叫的時機之整理：
  - init()　這是applet啟動時第一個呼叫的method，它只執行一次，主要是用來對applet設定初值之用
  - start()　呼叫完init() method之後，接著便立刻呼叫start()。只要applet的畫面出現一次，start() 便會被呼叫一次。如果切換到其它網頁瀏覽，再跳回本頁時，start() 仍會再執行一次
  - stop()　當切換到其它網頁瀏覽，或者是關閉瀏覽器時，便會執行stop() method以暫停applet的執行
  - destroy() 關閉瀏覽器時會先呼叫stop() 暫停執行applet，然後呼叫destroy()來釋放原先被applet佔去的記憶體空間

applet的生命週期

- 下面的流程圖繪出applet的生命週期：



觀察applet的生命週期 (1/2)

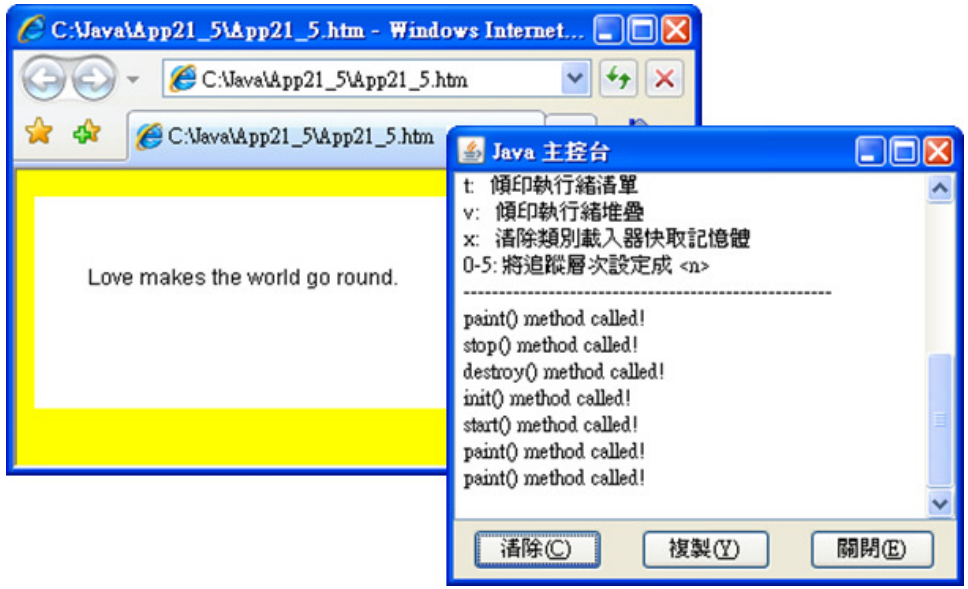
- App21_5會追蹤applet執行時，各method的生命週期

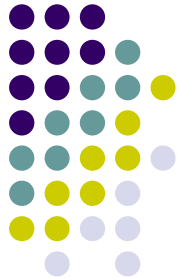
```
01  // App21_5, 簡單的 applet 程式
02  import java.awt.*;
03  import java.applet.Applet;
04
05  public class App21_5 extends Applet
06  {
07     public void init()                    // init() method
08     {
09        System.out.println("init() method called!");
10     }
11     public void start()                   // start() method
12     {
13        System.out.println("start() method called!");
14     }
15     public void stop()                    // stop() method
16     {
17        System.out.println("stop() method called!");
18     }
```

觀察applet的生命週期 (2/2)

```
19      public void destroy()                  // destroy() method
20      {
21          System.out.println("destroy() method called!");
22      }
23      public void paint(Graphics g)
24      {
25          g.drawString("Love makes the world go round.",30,50);
26          System.out.println("paint() method called!");
27      }
28  }
```

```
01  <!-- App21_5.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05      CODE     = "App21_5.class"
06      WIDTH    = "250"
07      HEIGHT   = "120"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```





加入AWT元件 (1/2)

- App21_6在applet視窗裡加入一個按鈕

```
01  // App21_6, 加入 AWT 元件到 applet 裡
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05
06  public class App21_6 extends Applet implements ActionListener
07  {
08     Button btn;                          // 宣告 Button 型態的變數 btn
09     public void init()
10     {
11        btn=new Button("Start");          // 建立 btn 物件
12        btn.addActionListener(this);      // 以 applet 本身當成 btn 的傾聽者
13        add(btn); // 將 btn 按鈕加入 applet 視窗裡
14     }
15     public void actionPerformed(ActionEvent e)
16     {
17        if(btn.getLabel()=="Start")
18           btn.setLabel("Stop");          // 設定按鈕上方的文字為 Stop
19        else
20           btn.setLabel("Start");         // 設定按鈕上方的文字為 Start
21     }
22  }
```

btn為「實例變數」

this指的就是applet本身

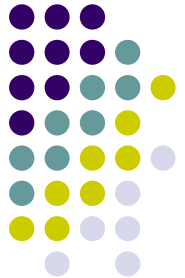
可以把13行改寫成：
`this.add(btn);`

加入AWT元件 (2/2)

```
01  <!-- App21_6.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05      CODE     = "App21_6.class"
06      WIDTH    = "180"
07      HEIGHT   = "120"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```

- App21_6在appletviewer視窗內執行的結果

按鈕btn的標題會在
Start與Stop之間變換



在applet視窗內繪圖 (1/2)

- App21_7是在applet內撰寫滑鼠事件的練習

```
01  // App21_7, 按滑鼠右鍵繪出圓形
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05
06  public class App21_7 extends Applet implements MouseListener
07  {
08     int x,y;                                  // 滑鼠指標的 x、y 座標
09     boolean clicked=false;
10
11     public void init()
12     {
13        this.addMouseListener(this);           // 設定 applet 為自己本身的傾聽者
14     }
15     public void mouseClicked(MouseEvent e)
16     {
17        clicked=true;
18        x=e.getX();                            // 取得滑鼠按下之點的 x 座標
19        y=e.getY();                            // 取得滑鼠按下之點的 y 座標
20        update(getGraphics());                 // 清除繪圖區，然後重繪
21     }
```

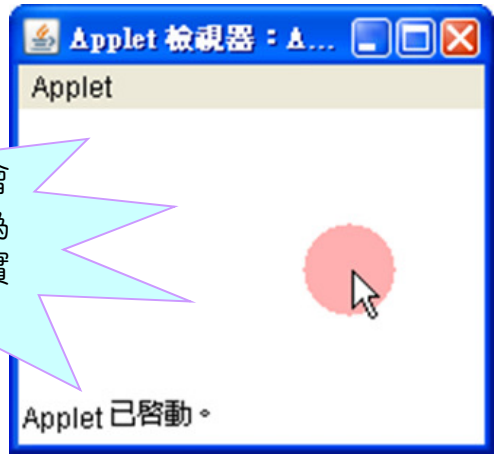
可改寫成
`addMouseListener(this);`

在applet視窗內繪圖 (2/2)

```
22      public void paint(Graphics g)
23      {
24        if(clicked)
25        {
26           g.setColor(Color.pink);          // 設定顏色為粉紅色
27           g.fillOval(x-20,y-20,40,40);     // 以按下的位置為圓心繪出圓形
28        }
29      }
30       public void mouseEntered(MouseEvent e){}
31       public void mouseExited(MouseEvent e){}
32       public void mousePressed(MouseEvent e){}
33       public void mouseReleased(MouseEvent e){}
34    }
```

```
01    <!-- App21_7.htm -->
02    <HTML>
03    <BODY BGCOLOR = "FFFF00" >
04    <APPLET
05       CODE     = "App21_7.class"
06       WIDTH    = "200"
07       HEIGHT   = "120"  >
08    </APPLET>
09    </BODY>
10    </HTML>
```

按下滑鼠按鈕時，會以指標箭頭的頂端為中心繪出粉紅色的實心圓



載入與顯示影像

- Image類別
  - 「影像」是由Image類別所建立的物件
  - Image類別置於java.awt類別庫
- 載入與顯示圖檔只要下列三個動作：
  - 宣告Image類別型態的變數
  - 利用getImage() 載入圖檔
  - 利用drawImage() 繪出圖檔

載入圖檔 (1/2)

- 下面的範例說明如何在applet裡載入一個jpg圖檔

```
01  // App21_8, 在 applet 裡載入圖檔
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05
06  public class App21_8 extends Applet
07  {
08     Image img;                          // 宣告 Image 類別型態的變數 img
09
10     public void init()
11     {
12        img=getImage(getCodeBase(),"flower.jpg");       // 載入圖檔
13     }
14
15     public void paint(Graphics g)
16     {
17        g.drawImage(img,20,20,this); // 將 img 畫在 applet 上
18     }
19  }
```

如果要限定載入後，圖形的寬度與高度，可改成
`g.drawImage(img,20,20,w,h,this);`

載入圖檔 (2/2)

```
01  <!-- App21_8.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05     CODE     = "App21_8.class"
06     WIDTH    = "300"
07     HEIGHT   = "200"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```

w設為150，
h設為150

移動影像 (1/3)

- App21_9是移動一隻載入的小天使影像：

```
01  // App21_9, 在 applet 裡移動圖檔
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05
06  public class App21_9 extends Applet implements MouseMotionListener,MouseListener
07  {
08     Image img;                    // 宣告 Image 類別型態的變數 img
09     int x=10,y=10,posX=10,posY=10,dx,dy;
10
11     public void init()
12     {
13        img=getImage(getCodeBase(),"angle.gif");        // 載入影像
14        addMouseListener(this);
15        addMouseMotionListener(this);
16     }
17     public void mousePressed(MouseEvent e)
18     {
19        dx=e.getX()-posX;          // 取得按下之點與基準點 x 方向之距離
20        dy=e.getY()-posY;          // 取得按下之點與基準點 y 方向之距離
21     }
```

移動影像 (2/3)

```
22      public void mouseDragged(MouseEvent e)
23      {
24         x=e.getX()-dx;            // 取得拖曳時，基準點的 x 座標
25         y=e.getY()-dy;            // 取得拖曳時，基準點的 y 座標
26         if(dx>0 && dx<82 && dy>0 && dy<87)     // 如果指標落在圖形上方
27         {
28            Graphics g=getGraphics();
29            update(g);             // 清空畫面為背景顏色，再呼叫 paint()
30         }
31      }
32      public void paint(Graphics g)
33      {
34         g.drawImage(img,x,y,this);              // 將 img 畫在 applet 上
35         posX=x;                                 // 更新基準點的 x 座標
36         posY=y;                                 // 更新基準點的 y 座標
37      }
38      public void mouseMoved(MouseEvent e){}
39      public void mouseReleased(MouseEvent e){}
40      public void mouseEntered(MouseEvent e){}
41      public void mouseExited(MouseEvent e){}
42      public void mouseClicked(MouseEvent e){}
43   }
```

移動影像 (3/3)

```
01  <!-- App21_9.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05      CODE     = "App21_9.class"
06      WIDTH    = "340"
07      HEIGHT   = "170"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```

按下滑鼠按鈕拖曳時所產生的動畫。注意左圖僅是動畫的示意圖，真正在拖曳時只有一個小天使出現

畫面閃爍的處理

- 處理畫面閃爍的步驟

  (1) 建立一個與applet視窗一樣大小的繪圖區

  (2) 把影像繪在這個繪圖區內

  (3) paint事件一發生，便把applet內繪圖區的內容顯示成預先畫好影像的繪圖區

- 下面的程式解決App21_9畫面閃爍的問題

解決畫面閃爍 (1/3)

下面的程式解決App21_9畫面閃爍的問題

```
01  // App21_10, 解決畫面閃爍的問題
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05
06  public class App21_10 extends Applet implements MouseMotionListener,MouseListener
07  {
08     Image img;                          // 宣告 Image 類別型態的變數 img
09     Image imgB;                         // 宣告 Image 類別型態的變數 imgB
10     Graphics gB;                        // 宣告 Graphics 類別型態的變數 gB
11
12     int x=10,y=10,posX=10,posY=10,dx,dy;
13
14     public void init()
15     {
16        img=getImage(getCodeBase(),"angle.gif");   // 載入影像
17        addMouseListener(this);
18        addMouseMotionListener(this);
19
20        imgB=createImage(getWidth(),getHeight());       // 建立 imgB 物件
21        gB=imgB.getGraphics();                  // 取得 imgB 物件的繪圖區
22     }
23     public void mousePressed(MouseEvent e)
24     {
25        dx=e.getX()-posX;                   // 取得按下之點與基準點 x 方向之距離
26        dy=e.getY()-posY;                   // 取得按下之點與基準點 y 方向之距離
27     }
28     public void mouseDragged(MouseEvent e)
29     {
30        x=e.getX()-dx;                      // 取得拖曳時，基準點的 x 座標
31        y=e.getY()-dy;                      // 取得拖曳時，基準點的 y 座標
```

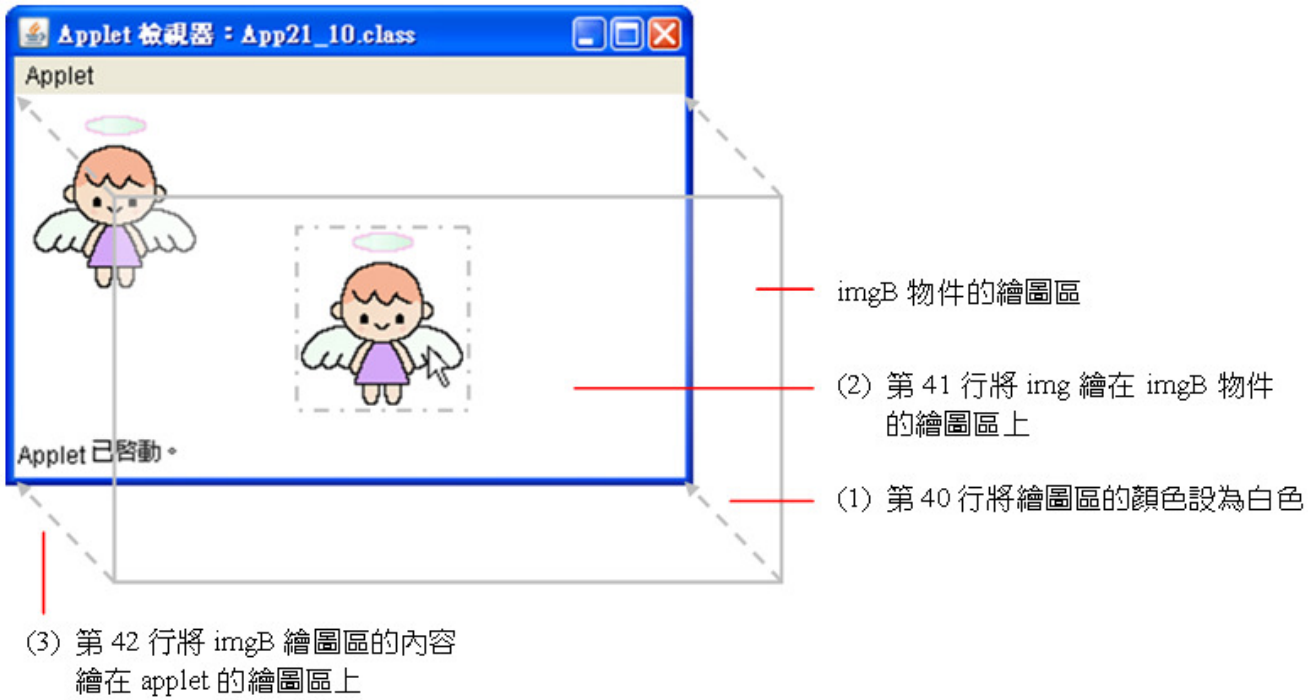
解決畫面閃爍 (2/3)

```
32          if(dx>0 && dx<82 && dy>0 && dy<87)     // 如果指標落在圖形上方
33          {
34             paint(getGraphics());               // 呼叫 paint() method
35          }
36       }
37       public void paint(Graphics g)
38       {
39          gB.setColor(new Color(255,255,255));      // 設定繪圖顏色為白色
40          gB.fillRect(0,0,getWidth(),getHeight()); // 以白色填滿整個畫面
41          gB.drawImage(img,x,y,this);               // 將 img 圖檔在 gB 中繪出
42          g.drawImage(imgB,0,0,this);    // 將 imgB 的內容顯示在 applet 上
43
44          posX=x;                        // 更新基準點的 x 座標
45          posY=y;                        // 更新基準點的 y 座標
46       }
47       public void mouseMoved(MouseEvent e){}
48       public void mouseReleased(MouseEvent e){}
49       public void mouseEntered(MouseEvent e){}
50       public void mouseExited(MouseEvent e){}
51       public void mouseClicked(MouseEvent e){}
52    }
```

```
01    <!-- App21 10.htm -->
02    <HTML>
03    <BODY BGCOLOR = "FFFF00" >
04    <APPLET
05       CODE     = "App21 10.class"
06       WIDTH    = "340"
07       HEIGHT   = "170"  >
08    </APPLET>
09    </BODY>
10    </HTML>
```

解決畫面閃爍 (3/3)

- 請參考下圖的解說，以便瞭解每一行程式碼的功用：



AudioClip介面的使用 (1/3)

- AudioClip介面可播放音樂，定義的method：

表 21.5.1　java.applet.AudioClip 的 method

method	主要功能
public void loop()	重複播放單曲音樂檔
public void play()	播放音樂檔
public void stop()	停止播放音樂檔

- 下面的程式是在applet播放音樂檔的範例

```
01  // App21_11, 在 applet 播放音樂檔
02  import java.awt.*;
03  import java.awt.event.*;
04  import java.applet.Applet;
05  import java.applet.AudioClip;       //  載入 AudioClip 類別
06
07  public class App21_11 extends Applet implements ItemListener
08  {
09     AudioClip midi[]=new AudioClip[3]; // 宣告 AudioClip 介面型態的陣列
10     AudioClip current;          // 宣告AudioClip介面型態的變數 current
```

AudioClip介面的使用 (2/3)

```
11
12      Choice chc=new Choice();          // 建立 Choice 元件
13
14      public void init()                                  載入三個聲音檔，並把它們
15      {                                                   設定給 midi 陣列
16         midi[0]=getAudioClip(getCodeBase(),"小叮噹.midi");
17         midi[1]=getAudioClip(getCodeBase(),"科學小飛俠.midi");
18         midi[2]=getAudioClip(getCodeBase(),"無敵鐵金剛.midi");
19         chc.add("小叮噹");
20         chc.add("科學小飛俠");
21         chc.add("無敵鐵金剛");
22         add(chc);
23         chc.addItemListener(this);    // 把 applet 當成 chc 的傾聽者
24         current=midi[0];              // 設定目前播放的歌曲為 midi[0]
25         current.play();               // 播放歌曲
26      }
27
28      public void itemStateChanged(ItemEvent e)
29      {
30         current.stop();                          // 停止播放歌曲
31         int index=chc.getSelectedIndex();        // 取得被選取的索引值
32         current=midi[index];          // 設定播放的歌曲為 midi[index]
33         current.play();               // 播放歌曲
34      }
35   }
```

AudioClip介面的使用 (3/3)

```
01  <!-- App21_11.htm -->
02  <HTML>
03  <BODY BGCOLOR = "FFFF00" >
04  <APPLET
05      CODE     = "App21_11.class"
06      WIDTH    = "180"
07      HEIGHT   = "100"  >
08  </APPLET>
09  </BODY>
10  </HTML>
```

AudioClip具有混聲的功能，也就是可以同時播放好幾個聲音檔

從下拉選單裡可以選擇欲播放的音樂