第 3 章　打字机式跑马灯系统

3.1　系 统 概 要

本系统是一个打字机式跑马灯系统，字体的出现类似于打字效果，并且具有超级链接的功能。

3.2　基本功能要求

① 在该系统中，用户可以在客户端浏览包含有 applet 的网页，打字机式跑马灯系统界面如图 3-1 所示。

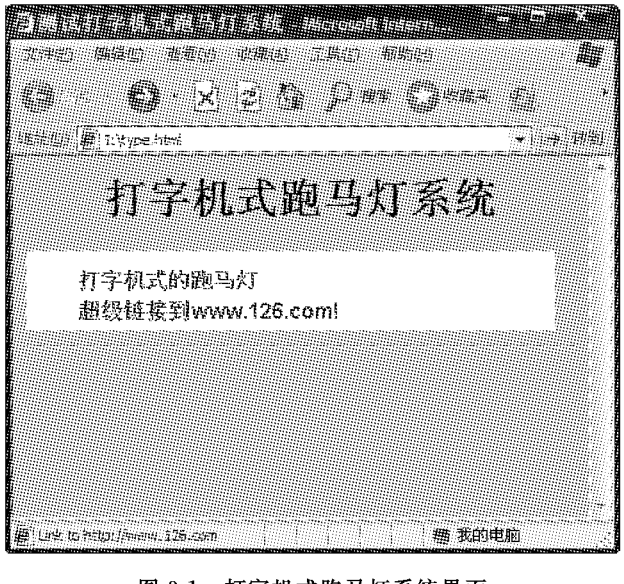

图 3-1　打字机式跑马灯系统界面

② 系统中包含有链接到所设定网址的超级链接，单击超级链接，进入相应的网站，如图 3-2 所示。

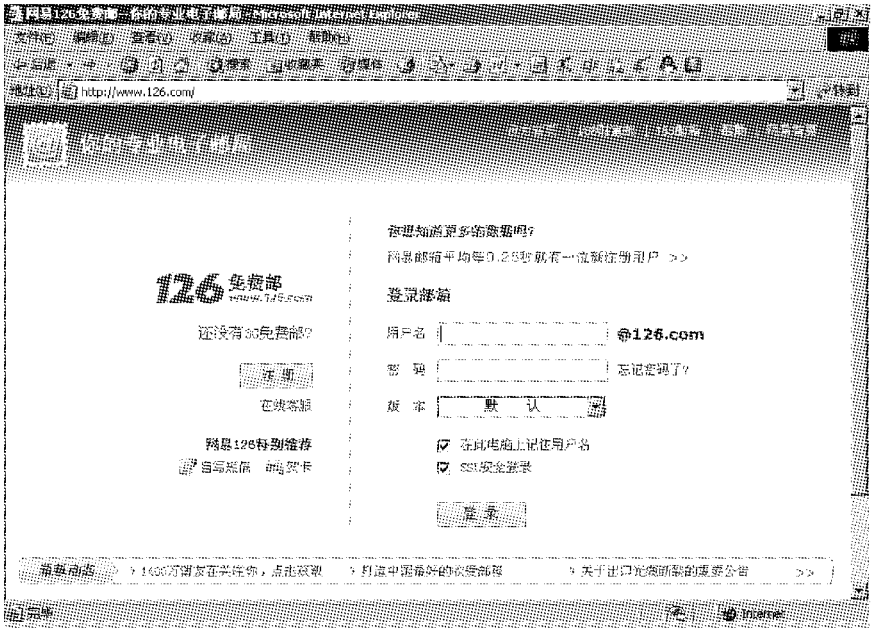

图 3-2　进入所链接的网站

3.3　主要知识点

用户所浏览的网页包含有 Applet，利用<applet></applet>引用该 Applet 所调用的类，即 WriterType.class。可以利用其中的 Param 属性对显示的文字及文字的相关属性进行设置。

类 WriterType 继承了 Applet 而且实现接口 Runnable。该类利用网页中 Param 传递的参数设置显示的文字及文字运动的相关属性。

3.4　系统设计思路

根据该系统需要实现的功能，网页中引入 Applet 所调用的类，并根据所要实现的效果设置不同的参数。类 WriterType 的实现过程中要根据网页中传来的参数，根据对应的方法设置显示的文字及文字运动的相关的属性。

3.5　部分源程序代码

(1) 程序分析

① Java Applet 就是用 Java 语言编写的一些应用程序，它们可以直接嵌入到网页中，并能够产生特殊效果。包含 Applet 的网页可以称其为 Java 支持的网页。当用户访问网

页时，Applet 被下载到用户的计算机上执行，但前提是用户使用的是支持 Java 的网络浏览器。通过浏览包含 Applet 的网页用户可以更好地欣赏网页上 Applet 产生的多媒体效果。

② 含有 Applet 网页的 HTML 文件代码中带有<applet>和</applet>这样一对标记。在网页中插入 Java Applet，它的格式为：

```
<applet code=java applet 的文件名 width=宽度 height=高度>
    <param name=参数名 value=参数值>
    <param name=参数名 value=参数值>
    …
</applet>
```

③ 在本程序中，字体出现类似于打字的效果，并且具有超级链接的功能。在进行参数设置时，可以进行设置的参数包括文字内容、文字字体、背景颜色、文字颜色、字体大小、文字速度、声音、文字显示区域的高、宽以及超级链接的地址等属性。

④ 对于要显示的文字内容，设定在固定宽度区域内显示。在 buildLines()方法内利用类 StringTokenizer 将要显示的文本内容转化为一组标记，通过 countTokens()方法得到转化后标记的个数，再使用 nextToken()将标记依次存储在 String 数组 words 里。再根据设定的宽度，将文字显示在固定区域内。

(2) WriterType.java 代码

```
1.  import java.applet.*;
2.  import java.awt.*;
3.  import java.io.*;
4.  import java.net.URL;
5.  import java.util.*;
6.
7.  // WriterType 类继承了 Applet 实现 Runnable 接口
8.  //在网页中利用 Applet 调用该类，实现打字机式跑马灯效果
9.  public class WriterType extends Applet implements Runnable {
10. // 定义相应的属性和参数，宽度、高度、背景颜色等
11. AudioClip hit, cr, ding;
12. boolean alreadyRun=false, soundOn=false, loop=false;
13. Color bgColor=Color.lightGray, textColor=Color.black;
14. Font font;
15. Image offScreen, background;
16. int width, height, currentline=0, currentword=0, currentletter=0,
17.           depth=0, margin=0, cycles=0, step=0, pause=0, speed=0,
18.           update=0, linecount=0;
19. long lastread=0;
20. MediaTracker mt;
21. String soundactivation, text, textfile, target;
22. Thread woohoo=null;
```

```
23. URL hotlink=null;
24. Vector lines=null;
25.
26. // 初始化参数
27. public WriterType() {
28.       alreadyRun=false;
29.       soundOn=false;
30.       loop=true;
31.       soundactivation="enter";
32.  }
33.
34.  // 根据所设定的每行文字的宽度改变所显示的文字显示方式
35. public void buildLines() {
36.       lines=new Vector();
37.       FontMetrics fontmetrics=offScreen.getGraphics().getFontMetrics
          ();
38.       StringTokenizer strTok=new StringTokenizer(text, "\n");
39.       while (strTok.hasMoreTokens()) {
40.           StringTokenizer strTok1=new StringTokenizer(strTok.nextToken
              ());
41.           int wordcount=strTok1.countTokens();
42.           String[] words=new String[wordcount];
43.           for (int i=0; i<wordcount; i++)
44.               words[i]=strTok1.nextToken();
45.
46.           String s="";
47.           for (int j=0; j<wordcount; j++) {
48.
49.             s=s!=null?s+words[j]+"":words[0];
50.             if (fontmetrics.stringWidth(s)>width-margin*2) {
51.                 lines.addElement(s.substring(0, s.lastIndexOf(" ",s
52.                       .lastIndexOf(" ")-1)));
53.                 s=words[j]+" ";
54.               }
55.       }
56.       lines.addElement(s);
57.       linecount=lines.size();
58.     }
59.
60.     depth=height-fontmetrics.getHeight()/2;
61.   }
62.
63.   // 读取跑马灯文字
64.   public void checkTextfile() {
```

```
65.      loop=true;
66.      text="";
67.      try {
68.        DataInputStream datainputstream=new DataInputStream((new URL(
69.            getDocumentBase(), textfile)).openStream());
70.        boolean flag=true;
71.        while (flag) {
72.          String s=datainputstream.readLine();
73.          if (s==null)
74.            flag=false;
75.          else
76.            text=text+s+"\n";
77.        }
78.
79.        datainputstream.close();
80.        lastread=(new Long((new Date()).getTime())).longValue();
81.        return;
82.      } catch (Exception exception) {
83.        System.out.println("OOOHH--"+exception.toString());
84.      }
85.    }
86.
87.    // 得到页面中传递的参数如 BGCOLOR、TEXTCOLOR 等,设置文字的属性
88.    public void init() {
89.      mt=new MediaTracker(this);
90.      lastread=0L;
91.      width=getSize().width;
92.      height=getSize().height;
93.      offScreen=createImage(width, height);
94.
95.      String param;
96.      if ((param=getParameter("BACKGROUND"))!=null) {
97.        try {
98.          background=getImage(new URL(getDocumentBase(), param));
99.        } catch (Exception e) {
100.        }
101.       if (background!=null)
102.         mt.addImage(background, 0);
103.     }
104.
105.     if ((param=getParameter("BGCOLOR"))!=null)
106.       bgColor=new Color(Integer.parseInt(param, 16));
107.
108.     if ((param=getParameter("TEXTCOLOR"))!=null)
```

```
109.       textColor=new Color(Integer.parseInt(param, 16));
110.
111.     String fontName="Helvetica";
112.     if ((param=getParameter("FONTNAME"))!=null)
113.       fontName=param;
114.     int fontSize=12;
115.     if ((param=getParameter("FONTSIZE"))!=null)
116.       fontSize=Integer.parseInt(param);
117.     int fontStyle=Font.PLAIN;
118.     if ((param=getParameter("FONTSTYLE"))!=null) {
119.       param=param.toUpperCase();
120.       if (param.indexOf("BOLD")!=-1)
121.         fontStyle |=Font.BOLD;
122.       if (param.indexOf("ITALIC")!=-1)
123.         fontStyle |=Font.ITALIC;
124.     }
125.     font=new Font(fontName, fontStyle, fontSize);
126.
127.     param=getParameter("CYCLES");
128.     if (param==null || param.equalsIgnoreCase("infinite")) {
129.       cycles=1;
130.       step=0;
131.     } else {
132.       cycles=Integer.parseInt(param);
133.       step=1;
134.     }
135.
136.     param=getParameter("MARGIN");
137.     margin=param==null?width/10:Integer.parseInt(param);
138.
139.     param=getParameter("PAUSE");
140.     pause=param==null?2000:Integer.parseInt(param);
141.
142.     param=getParameter("SOUNDACTIVATION");
143.     soundactivation=param==null?"enter": param.toLowerCase();
144.
145.     soundOn=soundactivation.equals("auto");
146.
147.     if ((param=getParameter("SOUND.KEYSTROKE"))!=null)
148.       try {
149.         hit=getAudioClip(new URL(getDocumentBase(), param));
150.       } catch (Exception e) {
151.       }
```

```
152.
153.      if ((param=getParameter("SOUND.RETURN"))!=null)
154.        try {
155.          cr=getAudioClip(new URL(getDocumentBase(), param));
156.        } catch (Exception e) {
157.        }
158.
159.      if ((param=getParameter("SOUND.BELL"))!=null)
160.        try {
161.          ding=getAudioClip(new URL(getDocumentBase(), param));
162.
163.        } catch (Exception _ex) {
164.        }
165.
166.      param=getParameter("SPEED");
167.      speed=param==null?100: Math.max(10, Integer.parseInt(param));
168.
169.      param=getParameter("TARGET");
170.      target=param==null?"_self" : param;
171.
172.      if ((param=getParameter("URL"))!=null)
173.        try {
174.          hotlink=new URL(getDocumentBase(), param);
175.        } catch (Exception e) {
176.        }
177.
178.      param=getParameter("TEXT");
179.      text=param==null?"This is a test...nthis is a test..." : param;
180.      text=text.replace('\\', '\n');
181.
182.      textfile=getParameter("TEXTFILE");
183.
184.      param=getParameter("UPDATE");
185.      update=param==null?15: Integer.parseInt(param);
186.
187.      buildLines();
188.      try {
189.        mt.waitForID(0);
190.        return;
191.      } catch (InterruptedException _ex) {
192.        return;
193.      }
194.    }
```

```
195.
196.   public void run() {
197.     currentline=0;
198.     for (int i=0; i<cycles; i+=step) {
199.       long l=(new Long((new Date()).getTime())).longValue();
200.       if (l-lastread>(long) (update*60000) && textfile!=null) {
201.         checkTextfile();
202.         buildLines();
203.       }
204.       for (int j=0; j<linecount; j++) {
205.         currentletter=1;
206.         String s=(String) lines.elementAt(j);
207.         for (int k=0; k<s.length(); k++) {
208.           if (soundOn && hit!=null)
209.             hit.play();
210.           if (k==s.length() && soundOn && cr!=null)
211.             cr.play();
212.           repaint();
213.           int i1=75+(int) (Math.random()*100D);
214.           try {
215.             Thread.sleep((i1*speed)/100);
216.           } catch (InterruptedException interruptedexception) {
217.             System.out.println("BB: "
218.                 +interruptedexception.toString());
219.           }
220.           currentletter++;
221.         }
222.
223.         currentletter=0;
224.         currentline++;
225.         alreadyRun=false;
226.
227.       }
228.       currentline=currentline%linecount;
229.       try {
230.         Thread.sleep(pause);
231.       } catch (InterruptedException interruptedexception) {
232.         System.out.println("AA: "+interruptedexception.toString());
233.
234.       }
235.     }
236.   }
237.
```

```
238.    // 设置背景颜色和字体颜色
239.    public void paintBuffer(Graphics g) {
240.      if (background!=null) {
241.        g.drawImage(background, 0, 0, this);
242.      } else {
243.        g.setColor(bgColor);
244.        g.fillRect(0, 0, width, height);
245.
246.      }
247.      g.setColor(textColor);
248.      g.setFont(font);
249.      FontMetrics fontmetrics=g.getFontMetrics();
250.      for (int i=0; i<currentline; i++)
251.        g.drawString((String) lines.elementAt(currentline-i-1), margin,
252.            depth-(i+1) * fontmetrics.getHeight());
253.
254.      String s=(String) lines.elementAt(currentline);
255.      String s1=currentletter>=s.length()?s:s.substring(0,
256.          currentletter);
257.      if (fontmetrics.stringWidth(s1)>(8*width)/10 &&!alreadyRun
258.          && soundOn) {
259.        alreadyRun=true;
260.        if (ding!=null)
261.          ding.play();
262.      }
263.      g.drawString(s1, margin, depth);
264.    }
265.
266.    public void paint(Graphics g) {
267.      paintBuffer(offScreen.getGraphics());
268.      g.drawImage(offScreen, 0, 0, this);
269.    }
270.
271.    public void update(Graphics g) {
272.      paint(g);
273.    }
274.
275.    // 启动线程
276.    public void start() {
277.      if (woohoo==null) {
278.        woohoo=new Thread(this);
279.        woohoo.start();
280.      }
```

```
281.    }
282.
283.    public void stop() {
284.      woohoo.stop();
285.      woohoo=null;
286.    }
287.
288.    //单击鼠标进入超级链接所指向的网址
289.    public boolean mouseEnter(Event event, int i, int j) {
290.      if (hotlink!=null)
291.        showStatus("Link to "+hotlink.toString());
292.      if (soundactivation.equals("enter"))
293.        soundOn=true;
294.      return true;
295.    }
296.
297.    public boolean mouseExit(Event event, int i, int j) {
298.      if (soundactivation.equals("enter"))
299.        soundOn=false;
300.      return true;
301.    }
302.
303.    public boolean mouseDown(Event event, int i, int j) {
304.      try {
305.        getAppletContext().showDocument(hotlink, target);
306.      } catch (Exception e) {
307.      }
308.      return true;
309.    }
310.  }
```

(3) Type.html 代码

```
1.  <html>
2.  <head>
3.  <title>
4.  测试打字机式跑马灯系统
5.  </title>
6.  </head>
7.  <body bgcolor="0ffff0">
8.  <h1 align="center">打字机式跑马灯系统</h1>
9.  <applet code="WriterType.class" width="400" height="60">
10. <param name="BGCOLOR" value="ffffff"/>
11. <param name="FONTSIZE" value="16"/>
```