Java Web 中随机汉字扭曲验证码的实现

常祖政　卢晓倩

摘　要：介绍验证码的工作原理，讨论了目前验证码存在的问题，提出了在 Java Web 中实现随机生成扭曲中文验证码的方法和具体的实验过程。

关键词：中文验证码；图像扭曲；网络安全

1　引言

验证码的英文 CAPTCHA，这个词最早是在 2002 年由卡内基梅隆大学的 Luis von Ahn、Manuel Blum、Nicholas J.Hopper 以及 IBM 的 John Langford 所提出。CAPTCHA 是“Completely Automated Public Turing test to tell Computers and Humans Apart”（全自动区分计算机和人类的图灵测试）的缩写，是一种区分用户是计算机和人的公共全自动程序。使用验证码可以防止对某些网站进行批量注册，重复发帖；防止他人使用广告软件发布大量的垃圾信息；防止密码被暴力破解，而且还可以防止对网站的恶意攻击 [1][2]。

2　验证码

2.1　工作原理

服务器端将随机生成的验证码字符串保存在内存中（一般是 Web 系统中的 session 对象），然后将该字符串写入图片，发送给浏览器端显示。在浏览器端，用户输人图片验证码上的字符串，然后提交给服务器端，服务器端将用户提交的字符串和服务器端保存在 session 对象中的字符串进行比较，判断是否一致，若一致就继续执行下面的代码段，从而完成后续操作，否则就无法使用后继功能。由于验证码是随机产生的，每次请求都会产生不同的字符串，攻击者无法从浏览器端提取出验证码信息，难以猜测其具体内容，这样就实现了阻挡攻击的目的 [2]。

2.2　验证码的种类及性能分析

目前使用的验证码主要有以下几种：纯字符验证码：由数字和英文字母构成，容易识别，便于输入，但是容易破解；图片验证码：将字符（主要是英文字符与数字）转换成图片，并且在其中加入干扰元素，人为识别较容易，机器识别难度增大，安全性比第一种要高；汉字验证码：使用汉字作为验证码字符，生成图片格式，加入干扰元素，机器识别难，安全性更高，是目前最新的；问题验证码：问题验证码主要是以问答式的形式来进行填写，目前使用较少。本文介绍的是第三种验证码——汉字验证码。

3　汉字编码原理

GB2312 或 GB2312-80 是一个简体中文字符集的中国国家标准，全称为《信息交换用汉字编码字符集·基本集》，由中国国家标准总局发布，1981 年 5 月 1 日实施。GB2312 编码通行于中国大陆，新加坡等地也采用此编码。中国大陆几乎所有的中文系统和国际化的软件都支持 GB2312。

GB2312 标准共收录 6763 个汉字，其中一级汉字 3755 个，二级汉字 3008 个；同时，GB2312 收录了包括拉丁字母、希腊字母、日文平假名及片假名字母、俄语西里尔字母在内的 682 个全角字符。

GB2312 中对所收汉字进行了“分区”处理，把七千余汉字以及标点符号、外文字母等，排成一个 94 行、94 列的方阵。方阵中每一横行叫一个“区”，每个区有九十四个“位”。一个汉字在方阵中的坐标称为该字的“区位码”。

（1）01-09 区为特殊符号。

（2）16-55 区为一级汉字，按拼音排序。

（3）56-87 区为二级汉字，按部首/笔画排序。

10-15 区及 88-94 区则未有编码。

举例来说，“啊”字是 GB2312 之中的第一个汉字，它的区位码就是 1601。

为了防止在验证码中出现偏难汉字，给验证工作带来麻烦，在进行生成汉字验证码时只使用 GB2312 字符集中的第一级常用汉字，共有 3755 个汉字。在《汉字区位码表》中第一级常用汉字在 16—55 区 [3]。

4　汉字扭曲效果验证码的实现

4.1　实现过程

如图 1 所示。

首先，随机生成规定字符数量的一级简码汉字，区码在 16——55 之间，位码在 1——93 之间，将此汉字区位码生成对应的汉字；然后将生成的汉字字符写入预先构建的图片中，可以添加干扰元素（背景干扰线或者干扰点，本文没有添加）；扭曲图片，目的很明确，提高机器 OCR 的难度，曾强安全性；

在扭曲的图片中添加干扰元素，文中使用的前景干扰点。

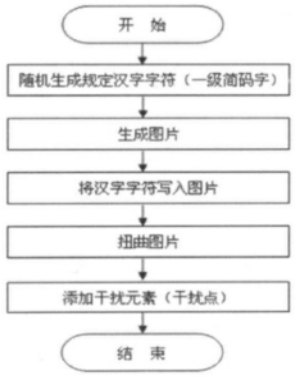

图 1

4.2 主要文件

如图 2 所示。

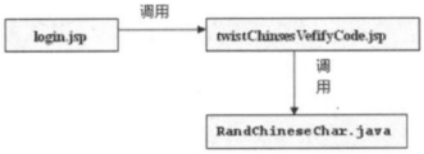

图 2

5 Java Web 中随机汉字扭曲验证码的实现代码

5.1 登录页面

运行 Login.jsp，界面如图所示，图 3 中显示出文中设计的验证码。

图 3

```
<% @ page contentType = "text/html; charset =gb2312" language="java" import="java.sql.*" errorPage="" %>
<head>
<! —客户端验证代码略——>
<meta http –equiv = "Content –Type" content = "text/html; charset=gb2312" />
<title>管理员登录</title>
<style type="text/css">
<! ––
th {
    background–color: #0080C0;
    border: 1px solid #0080C0;
    font–family: "宋体";
    font–size: 24px;
    color: #FFFF00;
}
.zi14px {
    font–size: 14px;
}
––>
</style>
</head>
<body>
<form id="loginForm" name="form1" method="post">
  <table width="400" border="0" align="center" cellpadding="0" cellspacing="0">
    <th height="30"  colspan="3">管理员登录 </th>
      <tr>
       <td  width = "76"  height = "30"  align = "right"  nowrap = "nowrap"  class="zi14px">帐号 :</td>
      <td height="30" colspan="2"  class="zi14px">
        <input name="userId" type="text" id="userId" />
      </td>
    </tr>
    <tr>
      <td height="30" align="right" nowrap="nowrap"  class="zi14px" >密码 :</td>
      <td height="30" colspan="2"  class="zi14px">
        <input name="pswd" type="password" id="pswd" />
      </td>
    </tr>
    <tr>
      <td height="25" align="right" nowrap="nowrap"  class="zi14px">验证码 :</td>
      <td  valign = "middle"  width = "179"  height = "25"  class = "zi14px">
       <input name="verifyCode" type="text" id="verifyCode" />
      </td>
       <td  valign = "middle"  class = "zi14px" ><img  src = "twistChinsesVefifyCode.jsp"/></td>
    </tr>
    <tr>
      <td height="25" colspan="3" nowrap="nowrap"  align="center"> </td>
    </tr>
    <tr>
   <td height="25" colspan="3"  align="center">
    <div align="center">
     <input type="submit" name="Submit" value="提交" />
          <input type="reset" name="Submit2" value="重置" />
           </div></td>
    </tr>
  </table>
</form>
</body>
</html>
```

5.2 彩色验证码

twistChinsesVefifyCode.jsp 的内容如下：

```
<%@ page pageEncoding ="gb2312" contentType ="image/jpeg" %>
<% @ page import = "com.sun.image.codec.jpeg.JPEGCodec"%>
<% @ page import = "com.sun.image.codec.jpeg.JPEGImageEncoder"%>
<%@ page import="test.*"%>
<%
    response.setHeader("Pragma","No-cache");
      response.setHeader("Cache-Control","no-cache");
      response.setDateHeader("Expires", 0);
      response.reset();
    int wordNum=2;//字符个数
   int fontSize=50;//字符大小(数字越大字越大)
     RandChineseChar randChineseChar =new RandChineseChar();//构造对象
     String randChChar =randChineseChar.getChineseChar(wordNum);//生成随机汉字字符
    session.setAttribute("verifyCode",randChChar);
// 放入 session
   session.setMaxInactiveInterval(60);
//设置 session 对象的生存期
    /* 得到输出流 */
      JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(response.getOutputStream());
   response.setContentType ("image/jpg"); encoder.encode(randChineseChar.getTwistGraphics (wordNum,fontSize,randChChar));
   out.clear();
   out = pageContext.pushBody();
%>
```

5.3 核心类

RandChineseChar.java 的内容如下：

```
package test;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.image.BufferedImage;
import java.util.Random;
public class RandChineseChar{
      private Random generator = new Random();
/**  随机产生定义的颜色  * */
private Color getRandColor() {
   Random random = new Random();
   Color color[] = new Color[10];
   color[0] = new Color(32, 158, 25);
   color[1] = new Color(218, 42, 19);
   color[2] = new Color(31, 75, 208);
    return new Color (random.nextInt (220), random.nextInt (220), random.nextInt(220));
}
//将随机生成的汉字区位码转换成对应的汉字,参数 code 是汉字区位码的字符型编码
public String getChineseCharByCode(int wordNum) {
   String chineseChar = "";
   String code=this.getWordCode(wordNum);
   for (int i = 0; i < code.length(); i += 4) {
        byte[] bytes = new byte[2];
        String lowCode = code.substring(i, i + 2);
        int tempLow = Integer.parseInt(lowCode);
        tempLow += 160;
        bytes[0] = (byte) tempLow;
        String highCode = code.substring(i + 2, i + 4);
        int tempHigh = Integer.parseInt(highCode);
        tempHigh += 160;
        bytes[1] = (byte) tempHigh;
        String chara = new String(bytes);
        chineseChar += chara;
    }
return chineseChar;
}
//随机产生一级简码(16—55 区)汉字的区位码,参数 wordNum 是汉字个数
 public String getWordCode(int wordNum){
    int i=1,j=0;
    int num;
    Random rand = new Random(System.currentTimeMillis());
     int     wordCodeByteNum=wordNum*2;
       int wordCodeArr[]=new int[wordCodeByteNum];
//存放区位码数字的数组
      //产生高位字节(区码)(16——55 范围区码)
            int wordHighByteScope=55;
            while (j<wordCodeByteNum)
            {
            i = rand.nextInt(wordHighByteScope);
            while(true){
                  if(i<16) i = rand.nextInt(wordHighByteScope);
                        else break ;
                       }
                        wordCodeArr[j]=i;
                        j+=2;
                  }
      //产生低位字节(位码)(1——93 范围位码)
      wordHighByteScope=93;
            j=1;
            while (j<wordCodeByteNum)
            {
            i = rand.nextInt(wordHighByteScope);
                  while(true){
                  if(i<16) i = rand.nextInt(wordHighByteScope);
                              else break ;
                             }
```

```
                    wordCodeArr[j]=i;
                    j+=2;
                }
  //将生成的区位码转换成字符型
                String wordCode=new String();
                for(i=0;i<wordCodeByteNum;i++){
                  wordCode+=wordCodeArr[i];
                }
                return wordCode;
  }
/* 实现图片扭曲开始 */
public Graphics shearGraphics (Graphics _g, int w1, int h1,
Color color) {
   Graphics g=_g;
   shearX(g, w1, h1, color);
   shearY(g, w1, h1, color);
   return g;
}
public void shearX(Graphics g, int w1, int h1, Color color) {
   int period = generator.nextInt(2);
   boolean borderGap = true;
   int frames = 1;
   int phase = generator.nextInt(2);
   for (int i = 0; i < h1; i++) {
    double d = (double) (period >> 1)
        * Math.sin ((double) i / (double) period +
(6.2831853071795862D * (double) phase)/ (double) frames);
     g.copyArea(0, i, w1, 1, (int) d, 0);
     if (borderGap) {
      g.setColor(color);
      g.drawLine((int) d, i, 0, i);
      g.drawLine((int) d + w1, i, w1, i);
     }
   }
}
public void shearY(Graphics g, int w1, int h1, Color color) {
   int period = generator.nextInt(40) + 10; // 50;
   boolean borderGap = true;
   int frames = 20;
   int phase = 7;
   for (int i = 0; i < w1; i++) {
    double d = (double) (period >> 1) * Math.sin((double) i /
(double) period + (6.2831853071795862D * (double) phase)
        / (double) frames);
     g.copyArea(i, 0, 1, h1, 0, (int) d);
     if (borderGap) {
      g.setColor(color);
      g.drawLine(i, (int) d, i, 0);
      g.drawLine(i, (int) d + h1, i, h1);
     }
   }
 }
```

```
/* 验证码图像扭曲结束 */
/* 生成验证码图片方法 */
public BufferedImage getTwistGraphics (int wordNum,int
fontSize,String randChar){
     //图片宽度和高度,有验证码汉字数量和字符大小决定
   int ImageWidth = fontSize*wordNum+10;
   int ImageHeight = fontSize+20;
     BufferedImage bi = new BufferedImage(ImageWidth +
20, ImageHeight,BufferedImage.TYPE_BYTE_INDEXED);
   Graphics graphics = bi.createGraphics();
   graphics.setColor(new Color(255,255,255));
   graphics.fillRect(0, 0, bi.getWidth(), bi.getHeight());
  /* 将字写到图片中 */
   graphics.setFont(new Font("全新硬笔楷书简", Font.BOLD,
fontSize));
   graphics.setColor(this.getRandColor());
  //将字写到图片中
    String randChineseChar =this.getChineseCharByCode
(wordNum);
    graphics.drawString(randChineseChar, 10,fontSize+5);
   //扭曲图形
   int w = bi.getWidth();
   int h = bi.getHeight();
   graphics=shearGraphics(graphics, w, h, Color.white);
   //画干扰点
  graphics.setColor(this.getRandColor());
   for(int i=0;i<ImageWidth*ImageHeight/5;i++){
      graphics.fill3DRect(10+generator.nextInt(ImageWidth),
generator.nextInt(ImageHeight),1,1,true);
  }
   return bi;
}
}
```

6 结语

文中讲述了汉字扭曲验证码的实现方法和具体实现代码，该方法能提高验证码的 OCR 识别难度，增强 Web 的安全性。文中代码经过严格调试，调试环境为 JDK1.6、Tomcat6.0。

参考文献

[1] http://baike.baidu.com/view/842.htm?fr=ala0_1_1.

[2] 黄赛平，许明. 验证码的识别与改进.南京师范大学学报（工程技术版），2009.7.

[3] http://baike.baidu.com/view/25492.htm.

[4] 陈占芳，冯欣，张伟. 随机中文汉字验证码的生产及其应用. 电脑知识与技术，2007.

[5] http://gundumw100.javaeye.com/blog/450948 .

[6] 褚建立，段雪丽. 基于 ASP.NET 的彩色汉字验证码的原理及其实现. 电脑知识与技术，2006.