中北大学软件学院

**实验报告**

|  |  |
| --- | --- |
| 专 业： | 软件工程 |
| 方 向： | 云计算与大数据分析 |
| 课程名称： | JavaEE框架与应用开发 |
| 班 级： | 15140Y01 |
| 学 号： | 1514010631 |
| 姓 名： | 杨晓辉 |
| 辅导教师： | 李华玲 |

2017年9月制

成绩：

|  |  |  |  |
| --- | --- | --- | --- |
| 实验时间 | 年 月 日 时至 时 | 学时数 |  |
| 1. 实验名称   熟悉Struts2核心文件 | | | |
| 1. 实验目的   (1)熟悉Struts2的配置文件web.xml和struts.xml。  (2)熟悉如何书写用户定义的控制器。  (3)学会struts2接收参数的方法。 | | | |
| 1. 实验内容   管理员登录后跳到管理员登陆成功的页面,普通用户登录后跳到普通用户登陆成功的页面,登录时用户名密码输入错跳到出错页注册和登录时用户名，密码没有输入时跳到登陆页，并在登陆页上显示校验失败的相关提示 | | | |
| 1. 文件描述   index.jsp 用户登陆页面 welcome.jsp 用户登陆后的欢迎页面  web.xml 服务器配置文件 struts.xml struts2配置文件  AdminLoginAction.kt/ UserLoginAction.kt 管理员登陆的action  CheckCodeAction.kt 验证码产生的action | | | |
| 1. 实验源代码   index.jsp 代码如下  <%@ taglib prefix="s" uri="/struts-tags" %> <%@ page contentType="text/html;charset=UTF-8" language="java" %> <html> <head>  <title>登陆页面</title>  <style>  .errorLabel {  font-weight: bold;  color: red;  }  </style>  <script>  function submitForm() {  var form = document.getElementById("form");  var userType = document.getElementById("userType");  var index = userType.selectedIndex;  var typeValue = userType.options[index].value;  if (typeValue === "管理员") {  form.action = "adminLogin.action"  } else if (typeValue === "普通用户") {  form.action = "userLogin.action"  } else {  alert("未知错误")  }  form.submit()  }  </script> </head> <body> <form method="post" id="form">   <table>  <tr>  <td>用户名</td>  <td><input type="text" name="username"></td>  <td></td>  <td class="errorLabel"><s:fielderror> <s:param>username</s:param></s:fielderror></td>  </tr>  <tr>  <td>密码</td>  <td><input type="password" name="password"></td>  <td></td>  <td class="errorLabel"><s:fielderror><s:param>password</s:param></s:fielderror></td>  </tr>  <tr>  <td>验证码</td>  <td><input type="text" name="checkCode"></td>  <td><img src="checkCode.action" onclick="this.src='checkCode.action'"></td>  <td class="errorLabel"><s:fielderror><s:param>code</s:param> </s:fielderror></td>  </tr>  <tr>  <td>用户类型</td>  <td>  <label for="userType"></label><select name="type" id="userType">  <option value="管理员">管理员</option>  <option value="普通用户">普通用户</option>  </select>   </td>  </tr>  <tr>  <td></td>  <td>  <input type="button" value="登陆" onclick="submitForm()"></td>  </tr>  </table> </form> </body> </html>    VerificationCode.kt代码:  class VerificationCode : ActionSupport() {    private var inputStream: ByteArrayInputStream? = null  private val WIDTH = 60  private val HEIGHT = 20   fun getInputStream(): ByteArrayInputStream? {  return inputStream  }   private fun setInputStream(inputStream: ByteArrayInputStream) {  this.inputStream = inputStream  }   private fun createRandom(): String {  val str = "0123456789qwertyupasdfghjkzxcvbnm"  val rands = CharArray(4)  val random = Random()  for (i in 0..3) {  rands[i] = str[random.nextInt(33)]  }  return *String*(rands)  }   private fun drawBackground(g: Graphics) {  g.*color* = Color(0xDCDCDC)  g.fillRect(0, 0, WIDTH, HEIGHT)  for (i in 0..119) {  val x = (Math.random() \* WIDTH).toInt()  val y = (Math.random() \* HEIGHT).toInt()  val red = (Math.random() \* 255).toInt()  val green = (Math.random() \* 255).toInt()  val blue = (Math.random() \* 255).toInt()  g.*color* = Color(red, green, blue)  g.drawOval(x, y, 1, 0)  }  }   private fun drawRands(g: Graphics, rands: String) {  g.*color* = Color.*BLACK* g.*font* = Font(null, *ITALIC* or *BOLD*, 18)  g.drawString("" + rands[0], 1, 17)  g.drawString("" + rands[1], 16, 15)  g.drawString("" + rands[2], 31, 18)  g.drawString("" + rands[3], 46, 16)  }   @Throws(Exception::class)  override fun execute(): String {  val response = ServletActionContext.getResponse()  response.setHeader("Pragma", "no-cache")  response.setHeader("Cache-Control", "no-cache")  response.setDateHeader("Expires", 0)  val rands = createRandom()  val image = BufferedImage(WIDTH, HEIGHT,  BufferedImage.*TYPE\_INT\_RGB*)   val g = image.*graphics*  drawBackground(g)  drawRands(g, rands)  g.dispose()  val outputStream = ByteArrayOutputStream()  ImageIO.write(image, "jpeg", outputStream)  val input = ByteArrayInputStream(outputStream  .toByteArray())  this.setInputStream(input)  ActionContext.getContext().*session*.put("checkCode", rands)  input.close()  outputStream.close()  return Action.*SUCCESS* } }  UserLogin.kt代码：  class UserLoginAction : ActionSupport() {   lateinit var username: String  lateinit var password: String  lateinit var checkCode: String   override fun validate() {  val code = ActionContext.getContext().*session*["checkCode"] as String  if (username.*trim*().*isEmpty*()) {  addFieldError("username", "用户名不能为空")  }  if (password.*trim*().*isEmpty*())  addFieldError("password", "密码不能为空")  if (!checkCode.*trim*().*equals*(code, true)) {  addFieldError("code", "验证码错误")  }  }  @Throws(Exception::class)  fun login(): String {  return if (username == "yxh" && password == "123") {  ActionContext.getContext().*session*.put("username", username)  ActionContext.getContext().*session*.put("type", "用户")  Action.*SUCCESS* } else {  ActionContext.getContext().put("error", "账号或密码错误")  Action.*ERROR* }  } } | | | |
| 1. 实验结论及心得   通过本次实验掌握了struts2的配置文件写法和自定义action。 | | | |