电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 2017221132009

姓 名 陆圣珩

（实验） 课程名称 面向对象程序设计（Java）

理论教师 周帆

实验教师 何中海

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：陆圣珩 学号：2017221302009 指导教师：周帆**

**实验地点：信软实验楼303 实验时间：2018.12.16**

1. **实验名称：Java高级编程技术实验**
2. **实验学时：2**
3. **实验目的：**

针对Java异常处理、多线程、I/O及网络以及GUI等Java高级技术的编程能力实验。

1. **实验原理：**

Java异常处理、多线程、I/O及网络以及GUI等Java高级技术。

1. **实验内容：**

1.完成第七章习题6编程。

2.完成第八章习题6编程。

3.完成第九章习题10编程。

4.完成第十章习题6编程。

5.完成第十一章习题7编程。

6.完成第十一章习题10编程。

1. **实验器材（设备、元器件）：**

PC机

1. **实验步骤：**

1、创建工程

2、编辑程序

3、编译程序

4、调试程序

5、运行程序，分析结果

本实验4个小题目都分别按上述步骤进行。

1. **实验结果与分析（含重要数据结果分析或核心代码流程分析）**

实验一

代码：

|  |
| --- |
| public class Expection {  public Expection() {  try {  try {  try {  int a[] = new int[2];  a[1]= 1/0;  a[4]= 1;  }catch(IndexOutOfBoundsException e) {  System.*err*.println("Exception msg:"+e.getMessage());  System.*err*.println("Exception string:"+e.toString());  e.printStackTrace();  }  }catch(ArithmeticException e) {  System.*err*.println("Exception msg:"+e.getMessage());  System.*err*.println("Exception string:"+e.toString());  e.printStackTrace();  }  }catch(NullPointerException e) {  System.*err*.println("Exception msg:"+e.getMessage());  System.*err*.println("Exception string:"+e.toString());  e.printStackTrace();  }  }  public static void main(String []args) {  new Expection();  } } |

运行结果：

|  |
| --- |
|  |

实验二

代码：

|  |
| --- |
| import java.time.\*;  class ClockThread implements Runnable{  String clock;  ClockThread(int i){  clock = "+0"+i+":00";  }  public void run(){  this.time();  }  synchronized void time() {  System.*out*.println(ZonedDateTime.now(ZoneId.of(this.clock)));  } }  class LocalClock extends ClockThread{  LocalClock() {  super(8);  }  public void run(){  this.sleep();  System.*out*.println(ZonedDateTime.now());  }  void sleep() {  try {  Thread.*sleep*(100);  } catch (InterruptedException e) {  e.printStackTrace();  }  } }  public class clock {  public static void main(String []args) {  ClockThread localtime = new LocalClock();  Thread thread = new Thread(localtime);  ClockThread time1 = new ClockThread(0);  Thread thread1 = new Thread(time1);  ClockThread time2 = new ClockThread(4);  Thread thread2 = new Thread(time2);  thread.start();  thread1.start();  thread2.start();  try {  thread.join();  thread1.join();  thread2.join();  } catch (InterruptedException e) {  e.printStackTrace();  }  } } |

运行结果：

|  |
| --- |
| Java 7 中不自带 java.time, 在java8以上的编译器中可以正常运行 |

实验三

代码：

|  |
| --- |
| **package** homework;  **import** java.io.\*;  **import** java.util.Scanner;  **public** **class** ReWrite {  **public** **static** **void** main(String[] args) {  PrintStream out;  **try** {  out = **new** PrintStream(**new** File("E:\\test.txt"));  Scanner sc = **new** Scanner(System.***in***);  String test = sc.nextLine();  out.print(test);  System.*setOut*(out);  } **catch** (FileNotFoundException e) {  e.printStackTrace();  }  }  } |

运行结果：

|  |
| --- |
|  |
|  |

实验四

代码：

|  |
| --- |
| import java.io.\*; import java.net.\*;  public class ClientChat {  public static final Integer *port* = 9001;  public static final String *host* = "127.0.0.1";   public static void main(String []args) {  Socket socket = null;  try {  socket = new Socket(*host*, *port*);  BufferedReader input = new BufferedReader(new InputStreamReader(socket.getInputStream()));  PrintStream out = new PrintStream(socket.getOutputStream());  System.*out*.println("ready send info to server!!");  out.println("HELLO WORLD!");  System.*out*.println("send info to server: "+ "HELLO WORLD!");  out.close();  input.close();  } catch (Exception e) {  System.*out*.println("ERROR");  } finally {  if (socket != null) {  try {  socket.close();  } catch (Exception e) {  socket = null;  }  }  }  } } |
| import java.io.BufferedReader; import java.io.IOException; import java.io.InputStreamReader; import java.io.PrintStream; import java.net.ServerSocket; import java.net.Socket;  public class ServerChat {   public static final Integer *port* = 9001;   public static void main(String[] args) {  try {  ServerSocket serverSocket = new ServerSocket(*port*);  while (true){  Socket socket = serverSocket.accept();  BufferedReader input = new BufferedReader(new InputStreamReader(socket.getInputStream()));  String clientContent = input.readLine();  System.*out*.println(clientContent);  PrintStream out = new PrintStream(socket.getOutputStream());  String s = new BufferedReader(new InputStreamReader(System.*in*)).readLine();  out.println("open");   out.close();  input.close();  serverSocket.close();  }  } catch (IOException e) {  System.*out*.println("ERROR: "+e.getMessage());  }  } } |

|  |
| --- |
|  |
|  |

实验五

代码：

|  |
| --- |
| import java.awt.Color; import java.awt.EventQueue; import java.awt.event.ActionEvent; import java.awt.event.ActionListener;  import javax.swing.JFrame; import javax.swing.JButton; import javax.swing.JLabel;  public class Button {   private JFrame frame;  private JLabel lblNewLabel;  private boolean flag = false;    public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  Button window = new Button();  window.frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }    public Button() {  initialize();  }   private void initialize() {  frame = new JFrame();  frame.setBounds(100, 100, 288, 234);  frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  frame.getContentPane().setLayout(null);   final JButton btnNewButton = new JButton("set red");  btnNewButton.setBounds(20, 30, 120, 30);  final JButton btnNewButton1 = new JButton("set green");  btnNewButton1.setBounds(20, 70, 120, 30);  final JButton btnNewButton2 = new JButton("set blue");  btnNewButton2.setBounds(20, 110, 120, 30);  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  btnNewButton.setBackground(Color.*red*);  btnNewButton1.setBackground(Color.*red*);  btnNewButton2.setBackground(Color.*red*);  }  });  btnNewButton1.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  btnNewButton.setBackground(Color.*green*);  btnNewButton1.setBackground(Color.*green*);  btnNewButton2.setBackground(Color.*green*);  }  });  btnNewButton2.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  btnNewButton.setBackground(Color.*blue*);  btnNewButton1.setBackground(Color.*blue*);  btnNewButton2.setBackground(Color.*blue*);  }  });  frame.getContentPane().add(btnNewButton);  frame.getContentPane().add(btnNewButton1);  frame.getContentPane().add(btnNewButton2);   lblNewLabel = new JLabel("");  lblNewLabel.setBounds(35, 107, 214, 36);  frame.getContentPane().add(lblNewLabel);  } } |

运行结果：

|  |
| --- |
|  |

实验六

代码：

|  |
| --- |
| import java.awt.\*; import java.awt.event.InputEvent; import java.awt.event.InputMethodEvent; import java.awt.event.KeyEvent; import javax.swing.\*; public class notepad extends JFrame{  *//文件组定义组件* JMenuBar jmb;*//菜单条组件* JMenu menu1,menu2,menu3,menu4,menu5;*//主菜单：文件、编辑、格式、查看、帮助* JMenuItem item2,item3,item4,item5,item6,item7;*//子菜单：新建、打开、保存、另存为、页面设置、打印、退出* JMenu xinjian;*//二级菜单* JMenuItem file,project;  *//编辑组定义组件  //编辑：撤消、剪切、复制、粘贴、删除、查找、查找下一个、替换、转到、全选、时间/日期* JMenuItem eitem1,eitem2,eitem3,eitem4,eitem5,eitem6,eitem7,eitem8,eitem9,eitem10,eitem11;   *//格式组定义组件* JMenuItem oitem1,oitem2;*//格式：自动换行、字体   //查看组定义组件* JMenuItem vitem1;*//查看：状态栏   //帮助组定义组件* JMenuItem hitem1,hitem2;*//帮助：查看帮助、关于记事本* JTextArea jta;  *//工具条* JToolBar jtb;  JButton jb1,jb2,jb3,jb4,jb5,jb6,jb7,jb8;   public static void main(String[] args) {  notepad win=new notepad();  }  *//构造函数* public notepad(){  *//构建组件  //工具条* jtb=new JToolBar();  jb1=new JButton(new ImageIcon("images/new.jpg"));  jb1.setToolTipText("新建");*//设置提示信息* jb2=new JButton(new ImageIcon("images/open.jpg"));  jb2.setToolTipText("打开");  jb3=new JButton(new ImageIcon("images/save.jpg"));  jb3.setToolTipText("保存");  jb4=new JButton(new ImageIcon("images/copy.jpg"));  jb4.setToolTipText("复制");  jb5=new JButton(new ImageIcon("images/delete.jpg"));  jb5.setToolTipText("删除");  jb6=new JButton(new ImageIcon("images/modify.jpg"));  jb6.setToolTipText("编辑");  jb7=new JButton(new ImageIcon("images/print.jpg"));  jb7.setToolTipText("打印");  jb8=new JButton(new ImageIcon("images/close.jpg"));  jb8.setToolTipText("关闭");   jmb=new JMenuBar();  *//主菜单* menu1=new JMenu("文件(F)");  menu1.setMnemonic('F');*//设置助记符* menu2=new JMenu("编辑(E)");  menu2.setMnemonic('E');  menu3=new JMenu("格式(O)");  menu3.setMnemonic('O');  menu4=new JMenu("查看(V)");  menu4.setMnemonic('V');  menu5=new JMenu("帮助(H)");  menu5.setMnemonic('H');   *//文件--新建--子目录* xinjian=new JMenu("新建");  file=new JMenuItem("文件");  project=new JMenuItem("工程");   item2=new JMenuItem("打开(O)");  item2.setMnemonic('O');  item2.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_O*,InputEvent.*CTRL\_MASK*));*//设置ctrl快捷组合键* item3=new JMenuItem("保存(S)");  item3.setMnemonic('S');  item3.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_S*,InputEvent.*CTRL\_MASK*));  item4=new JMenuItem("另存为(A)");  item4.setMnemonic('A');  item5=new JMenuItem("页面设置(U)");  item5.setMnemonic('U');  item6=new JMenuItem("打印(P)");  item6.setMnemonic('P');  item6.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_P*,InputEvent.*CTRL\_MASK*));  item7=new JMenuItem("退出(X)");  item7.setMnemonic('X');   eitem1=new JMenuItem("撤消(U)");  eitem1.setMnemonic('U');  eitem1.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_U*,InputEvent.*CTRL\_MASK*));  eitem2=new JMenuItem("剪切(T)");  eitem2.setMnemonic('T');  eitem2.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_T*,InputEvent.*CTRL\_MASK*));  eitem3=new JMenuItem("复制(C)");  eitem3.setMnemonic('C');  eitem3.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_C*,InputEvent.*CTRL\_MASK*));  eitem4=new JMenuItem("粘贴(P)");  eitem4.setMnemonic('P');  eitem4.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_P*,InputEvent.*CTRL\_MASK*));  eitem5=new JMenuItem("删除(L)");  eitem5.setMnemonic('L');  eitem5.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_DELETE*,0));  eitem6=new JMenuItem("查找(F)");  eitem6.setMnemonic('F');  eitem6.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_F*,InputEvent.*CTRL\_MASK*));  eitem7=new JMenuItem("查找下一个(N)");  eitem7.setMnemonic('N');  eitem7.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_F3*,0));  eitem8=new JMenuItem("替换(R)");  eitem8.setMnemonic('R');  eitem8.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_R*,InputEvent.*CTRL\_MASK*));  eitem9=new JMenuItem("转到(G)");  eitem9.setMnemonic('G');  eitem9.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_G*,InputEvent.*CTRL\_MASK*));  eitem10=new JMenuItem("全选(A)");  eitem10.setMnemonic('A');  eitem10.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_A*,InputEvent.*CTRL\_MASK*));  eitem11=new JMenuItem("时间/日期(D)");  eitem11.setMnemonic('D');  eitem11.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.*VK\_F5*,0));   oitem1=new JMenuItem("自动换行(W)");  oitem1.setMnemonic('W');  oitem2=new JMenuItem("字体(F)");  oitem2.setMnemonic('F');   vitem1=new JMenuItem("状态栏(S)");  vitem1.setMnemonic('S');   hitem1=new JMenuItem("查看帮助(H)");  hitem1.setMnemonic('H');  hitem2=new JMenuItem("关于记事本(A)");  hitem2.setMnemonic('A');   jta=new JTextArea();   *//设定布局管理器   //加入组件  //将按钮添加到工具条上* jtb.add(jb1);  jtb.add(jb2);  jtb.add(jb3);  jtb.add(jb4);  jtb.add(jb5);  jtb.add(jb6);  jtb.add(jb7);  jtb.add(jb8);   *//将菜单项添加到菜单上* xinjian.add(file);  xinjian.add(project);   menu1.add(xinjian);  menu1.add(item2);  menu1.add(item3);  menu1.add(item4);  menu1.addSeparator();*//添加分割线* menu1.add(item5);  menu1.add(item6);  menu1.addSeparator();  menu1.add(item7);   menu2.add(eitem1);  menu2.addSeparator();  menu2.add(eitem2);  menu2.add(eitem3);  menu2.add(eitem4);  menu2.add(eitem5);  menu2.addSeparator();  menu2.add(eitem6);  menu2.add(eitem7);  menu2.add(eitem8);  menu2.add(eitem9);  menu2.addSeparator();  menu2.add(eitem10);  menu2.add(eitem11);   menu3.add(oitem1);  menu3.add(oitem2);   menu4.add(vitem1);   menu5.add(hitem1);  menu5.addSeparator();  menu5.add(hitem2);   *//将菜单添加到菜单条上* jmb.add(menu1);  jmb.add(menu2);  jmb.add(menu3);  jmb.add(menu4);  jmb.add(menu5);   *//将菜单条添加到窗体上* this.setJMenuBar(jmb);   *//将工具条添加到窗体上* this.add(jtb,BorderLayout.*NORTH*);   JScrollPane jsp=new JScrollPane(jta);  jsp.setVerticalScrollBarPolicy(JScrollPane.*VERTICAL\_SCROLLBAR\_ALWAYS*);  this.add(jsp);   *//窗体设置* this.setTitle("notepad--");  this.setSize(800, 500);  this.setLocationRelativeTo(null);  this.setVisible(true);  this.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  } } |

运行结果：

|  |
| --- |
|  |

**九．总结及心得体会：**

相比前几次实验，本次实验的内容更加贴近实际生活的使用，在工程项目编程时，相比算法等更注重内在逻辑而非java本身的业务能力，java强大的包和框架会更加重要，本次实验就很好的展示了这一点，虽然部分包和代码显得有些过时，但这有助于初学者更好的理解并掌握。

**十．对本实验过程及方法、手段的改进建议：**

需注意Java版本的问题，以防出现必要的包无法使用的问题。

**报告评分：**

**指导教师签字：**