## 中国矿业大学计算机学院

**级本科生课程设计报告**

课程名称

报告时间

学生姓名

学 号

专 业

任课教师 薛猛

## 任课教师评阅

|  |  |  |
| --- | --- | --- |
|  | | |
| 设计报告成绩（40%） | 软件代码成绩（60%） | 总成绩 |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 教师签名 |  | 日 期 |  |

## 目 录

**实验一 简单计算器**

**1 系统概述**

本项目是一个简易计算器，可以实现常规的加减法、乘除法、取余、求倒数、求根、平方等混合运算，支持实数运算，满足基本的运算需求。

**2 系统设计**

**2.1 设计目标**

完成形如 2+100\*3.5-5^-sqrt(36) 的基本运算。

即可以连续运算，符合运算符计算优先级。

**2.2 设计分析与算法流程**

本项目主要采用了栈和字符串扫描来实现运算功能，有符号栈和值栈，符号栈保存已经扫描过的符号，值栈保存中间计算的结果，从左到右扫描用户输入的运算表达式，如果遇到值就将值压入值栈，遇到符号就判断栈顶的符号的优先级和栈外符号的优先级。

**关键点1：记录输入实数的值**

用一个字符串保存数字的值，从左到右扫描用户输入，当遇到数字和小数点时，将数字字符串进行字符串连接，保存之前输入的数字，遇到第一个运算符时，将最后的字符串结果转换为数字，然后压入值栈中，同时将字符串变为空串，以便记录下一个数。

**关键点2：解决运算符优先级**

1. **若栈顶符号优先级>=栈外符号优先级**

弹出符号栈顶的符号t，并连续弹出两个值栈的栈顶值v1,v2,并根据t做相应的运算，运算结果result压入值栈，同时将栈外符号压入符号栈内。

1. **若栈顶优先级<栈外优先级**

直接将栈外的运算符压入栈内，继续扫描下一个字符。（此时栈顶运算符不能计算）

**关键点3**：对平方、求根、求倒等运算的处理

若遇到平方^、求根s、求倒d等运算符号,则立即进行运算。

**2.3 界面设计**



图2-3-1

**3 系统实现（运行调试**）

package operation;

public class Operation extends JFrame {

public static void main(String[] args) {

// Create application frame.

Operation frame = new Operation();

frame.setTitle("Calculator");

frame.setSize(240, 320);

frame.setResizable(false);

frame.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

frame.init();

// Show frame

frame.setVisible(true);

}

private void init()

{

charStack=new Stack();

valueStack=new Stack<Double>();

charStack.push('#');

textField = new JTextField();

textField.setEditable(false);

textField.setHorizontalAlignment (JTextField.RIGHT);

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(5, 5));

Container container = getContentPane();

container.add(textField, BorderLayout.NORTH);

container.add(useButton('#'),BorderLayout.SOUTH);

container.add(panel, BorderLayout.CENTER);

panel.add(useButton('7'));

panel.add(useButton('8'));

panel.add(useButton('9'));

panel.add(useButton('+'));

//panel.add(useButton('s'));

panel.add(useButton('4'));

panel.add(useButton('5'));

panel.add(useButton('6'));

panel.add(useButton('-'));

// panel.add(useButton('d'));

panel.add(useButton('1'));

panel.add(useButton('2'));

panel.add(useButton('3'));

panel.add(useButton('\*'));

panel.add(useButton('0'));

panel.add(useButton('.'));

panel.add(useButton('='));

panel.add(useButton('/'));

panel.add(useButton('s'));

panel.add(useButton('d'));

panel.add(useButton('^'));

panel.add(useButton('%'));

}

public JButton useButton (final char key)

{

JButton button = new JButton(String.valueOf(key));

button.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent event)

{

JButton btn = (JButton) event.getSource();

char key2 = btn.getText().charAt(0);

action(key2);

}

});

button.addKeyListener(new KeyAdapter()

{

public void keyReleased(KeyEvent event)

{

char key2 = event.getKeyChar ();

action(key2);

}

});

return button;

}

private void operate(char key2){

v="";

String t=charStack.peek().toString();

double v1=0;

double v2=0;

double result=0;

if(key2=='+'||key2=='-'){

switch(t){

case "+":

v1=valueStack.pop();

v2=valueStack.pop();

result=v1+v2;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "-":

v1=valueStack.pop();

v2=valueStack.pop();

result=v2-v1;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "\*":

v1=valueStack.pop();

v2=valueStack.pop();

result=v1\*v2;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "/":

v1=valueStack.pop();

v2=valueStack.pop();

result=v2/v1;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "%":

v1=valueStack.pop();

v2=valueStack.pop();

result=v2%v1;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "#":

charStack.push(key2);

}

}else if(key2=='\*'||key2=='/'||key2=='%'){

switch(t){

case "+":

charStack.push(key2);

break;

case "-":

charStack.push(key2);

break;

case "\*":

v1=valueStack.pop();

v2=valueStack.pop();

result=v1\*v2;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "/":

v1=valueStack.pop();

v2=valueStack.pop();

result=v2/v1;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "%":

v1=valueStack.pop();

v2=valueStack.pop();

result=v2%v1;

valueStack.push(result);

charStack.pop();

charStack.push(key2);

break;

case "#":

charStack.push(key2);

}

}else if(key2=='d'||key2=='s'||key2=='^'){

v1=valueStack.pop();

if(key2=='d'){

result=1.0/v1;

}else if(key2=='s'){

result=Math.sqrt(v1);

}else{

result=Math.pow(v1, 2);

}

v=result+"";

}else if(key2=='='){

String op=charStack.pop().toString();

while(!op.equals("#")){

v1=valueStack.pop();

v2=valueStack.pop();

if(op.equals("+")){

result=v1+v2;

}else if(op.equals("-")){

result=v2-v1;

}else if(op.equals("/")){

result=v2/v1;

}else if(op.equals("\*")){

result=v2\*v1;

}else if(op.equals("%")){

result=v2%v1;

}

valueStack.push(result);

op=charStack.pop().toString();

}

charStack.push('#');

}

}

private void action(char key2)

{

switch(key2)

{

case '+':

case '-':

case 's':

case 'd':

case '^':

case '\*':

case '%':

case '/':

//if(v!=""){

if(sign==true){

v=tempResult;

sign=false;

}

valueStack.push(Double.parseDouble(v));

//}

operate(key2);

break;

case '=':

valueStack.push(Double.parseDouble(v));

operate(key2);

Double value=valueStack.pop();

tempResult=value.toString();

textField.setText(tempResult);

charStack.push('#');

sign=true;

break;

case '#':

textField.setText("");

v="";

sign=false;

charStack.removeAllElements();

charStack.push('#');

valueStack.removeAllElements();

break;

case '1':

case '2':

case '3':

case '4':

case '5':

case '6':

case '7':

case '8':

case '9':

case '0':

case '.':

v=v+String.valueOf(key2);

textField.setText(String.valueOf(v));

System.out.println(v);

break;

default:;

}

}

private JTextField textField;

private Stack charStack;

private Stack<Double> valueStack;

private String v="";

private char operator;

private boolean sign=false;

private String tempResult;

}

检验表达式2+100\*3.5-5^-sqrt(36)，先点击带有数值2的按钮，然后依次点击+、1、0、0、\*、3、. 、5等按钮可以看到结果等于352.0。



图3-1 输入表达式2+100\*3.5

继续点击-、5、^得到值327.0。

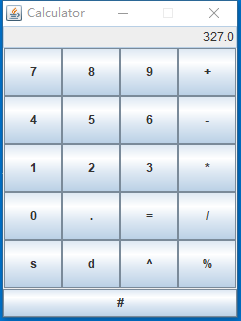


图3-2 输入表达式-5^

最后点击-、s、36的到最终结果321.0，检验结果正确。

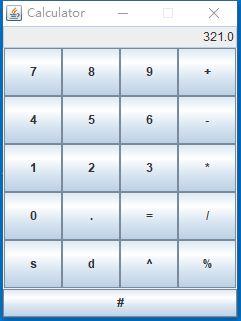


图3-3 输入表达式-sqrt(36)

结论: 计算器满足加减乘除运算优先级，满足连续的四则运算功能，可以进行连续运算，算法设计正确。

**4 系统扩展**

扩展了求倒数、求平方差、求余数以及求平方等功能。

**5 总结**

本实验加深了我对数据结构中栈的理解，关于运算符的优先级如何设置，计算顺序如何确定，都要通过栈来实现，优先级低的运算符压入符号栈后不立即出栈，等待后续的运算符进行进一步判断，优先级高的运算符立即出符号栈计算，操作数都压入数据栈，做运算时，根据运算符的操作数个数弹出相应的操作数，然后结果压入数据栈。总的来说，收获不小，编程能力得到了提高。

**实验二 学生通讯录**

**1 系统概述**

该项目目标是主要内容是设计开发一个学生通讯录软件，该学生通讯录采用xml格式存储数据，可导出excel格式文件，具备学生信息添加、编辑、删除、查找等功能。设计过程包括项目分析、界面设计、代码编写和运行调试。

**2 系统设计**

**2.1 设计目标**

设计开发一个学生通讯录软件，该学生通讯录采用xml格式存储数据，具备学生信息添加、编辑、删除、查找等功能。

**2.2 设计分析与算法流程**

各模块的功能描述如下：

学生信息添加：添加学生信息，包括姓名、学号、性别、年龄、出生日期、手机号、住址、专业等信息，不允许添加重复信息。

学生信息编辑：对存在的信息进行编辑、更新，编辑界面保存有学生已有的信息，无需重新输入。

学生信息删除：删除学生通讯录的学生信息，删除后无法恢复，选中一条学生信息进行删除。

学生信息查找：可根据学生通讯录中的学生学号查询学生信息，方便检索。

学生信息分类 :预先设置四类类别标签，对学生进行分类，可以分类管理学生信息。

学生信息备份：对保存学生的信息进行Excel文件格式导出，可以方便备份和统计。

图2-2-1 功能设计

**2.3 界面设计**



图2-3-1 界面设计

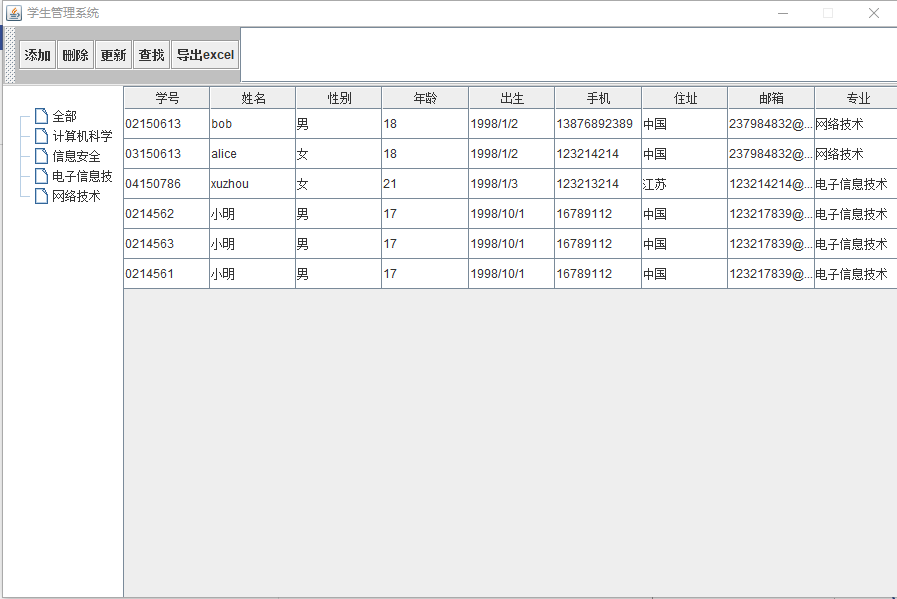


图2-3-2 主界面

**2.4 关键类图**

**3 系统实现（运行调试）**

package student;

Import ......;

public class StudentsFrame extends JFrame {

private static final LayoutManager FlowLayout = null;

public StudentsFrame() {

....(略）

getStudents();

}

private void action(String key){

if(key=="添加"){

showJD(key,selectRow);

}else if(key=="确认添加"){

if(checkEmpty(jts)) addStudent();

}else if(key=="确认更新"){

updateStudent(selectRow);

}else if(key=="更新"){

showJD(key,selectRow);

}else if(key=="查找"){

findStudent(findJt.getText());

}else if(key=="删除"){

deleteStudent(selectRow);

}else if(key=="导出excel"){

exportExcel();

}

}

private void findStudent(String id){

String searchModel[][]=new String[1][9];

for(int i=0;i<model.length;i++){

if(model[i][0].equals(id)) searchModel[0]=model[i];

}

updateView(searchModel);

}

private void updateStudent(int row){

if(row==-1) return;

String id=table.getModel().getValueAt(row, 0).toString();

NodeList idNode=document.getElementsByTagName("id");

Element newStudent=document.createElement("student");

Node oldStudent=getIdNode(id);

if(getIdNode(jts[0].getText())==null||id.equals(jts[0].getText())){

for(int i=0;i<attributesXML.length;i++){

Element e=document.createElement(attributesXML[i]);

Text t=document.createTextNode(jts[i].getText());

e.appendChild(t);

newStudent.appendChild(e);

}

oldStudent.getParentNode().replaceChild(newStudent, oldStudent);

writeDocument("test.xml");

getStudents();

selectRow=-1;

}

}

private void deleteStudent(int row){

if(row==-1) return;

String id=table.getModel().getValueAt(row, 0).toString();

NodeList idNode=document.getElementsByTagName("id");

Node student=getIdNode(id);

student.getParentNode().removeChild(student);

writeDocument("test.xml");

getStudents();

selectRow=-1;

}

private void showJD(String key,int row){

if(key=="添加"){

for(int i=0;i<attributes.length;i++){

jts[i].setText("");

}

}else if(key=="更新"){

if(row==-1) return;

for(int i=0;i<attributes.length;i++){

jts[i].setText(table.getModel().getValueAt(row, i).toString());

}

}

jd.setVisible(true);

}

public void exportExcel(){

ExcelExporter exporter=new ExcelExporter();

File xls=new File("test.xls");

try{

exporter.exportTable(table, xls);

}catch(IOException e){

}

}

public void writeDocument(String path){

try {

TransformerFactory tFactory =TransformerFactory.newInstance();

Transformer transformer = tFactory.newTransformer();

DOMSource source = new DOMSource(document);

StreamResult result = new StreamResult(new java.io.File("test.xml"));

transformer.transform(source, result);

}catch(TransformerException e){

}

}

public boolean checkEmpty(JTextField []jts){

for(int i=0;i<jts.length;i++){

if(jts[i].getText().equals("")) return false;

}

return true;

}

private void addStudent(){

if(getIdNode(jts[0].getText())!=null) return;

Element student=document.createElement("student");

for(int i=0;i<attributesXML.length;i++){

Element e=document.createElement(attributesXML[i]);

Text t=document.createTextNode(jts[i].getText());

e.appendChild(t);

student.appendChild(e);

}

document.getElementsByTagName("department").item(0).appendChild(student);

writeDocument("test.xml");

getStudents();

}

public Node getIdNode(String id){

NodeList idNode=document.getElementsByTagName("id");

for(int i=0;i<idNode.getLength();i++){

Node temp=idNode.item(i);

Node father=temp.getParentNode();

if(idNode.item(i).getFirstChild().getNodeValue().toString().equals(id)){

return father;

}

}

return null;

}

private void createJD(){

JPanel pan = new JPanel();

jd = new JDialog();

//pan.setLayout(FlowLayout);

for(int i=0;i<attributes.length;i++){

//if(i!=8){

pan.add(new JLabel(attributes[i]));

jts[i]=new JTextField(30);

pan.add(jts[i]);

//}else{

//}

}

JComboBox box=new JComboBox(majors);

box.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e){

jts[8].setText(box.getSelectedItem().toString());

}

});

pan.add(box);

pan.add(useButton("确认添加"));

pan.add(useButton("确认更新"));

jd.add(pan);

jd.setBounds(200,200,400,400);

//jd.setVisible(true);

}

public void useJTree(JTree tree){

tree.addMouseListener(new MouseAdapter(){

public void mouseClicked(MouseEvent e){

jtreeAction(e);

}

});

}

public void useTable(JTable t){

t.addMouseListener(new MouseAdapter(){

public void mouseClicked(MouseEvent e){

tableAction();

}

});

}

private void jtreeAction(MouseEvent e){

TreePath path=jtree.getPathForLocation(e.getX(), e.getY());

TreeNode node=(TreeNode)path.getLastPathComponent();

String major=node.toString();

ArrayList <String[]>students=new ArrayList();

String newModel[][];

int num=0;

if(!major.equals("全部")){

for(int i=0;i<model.length;i++){

if(model[i][8].equals(major)){

students.add(model[i]);

num++;

}

}

newModel=new String[num][attributes.length];

for(int j=0;j<newModel.length;j++){

newModel[j]=students.get(j);

}

updateView(newModel);

}else{

updateView(model);

}

}

private void tableAction(){

selectRow=getRowClicked();

}

public int getRowClicked(){

return table.getSelectedRow();

}

public JButton useButton (final String key)

{

JButton button = new JButton(String.valueOf(key));

button.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent event)

{

JButton btn = (JButton) event.getSource();

action(key);

}

});

button.addKeyListener(new KeyAdapter()

{

public void keyReleased(KeyEvent event)

{

char key2 = event.getKeyChar ();

action(key);

}

});

return button;

}

public static void main(String []args){

StudentsFrame myFrame=new StudentsFrame();

}

private void updateView(Object[][] newModel){

table.setModel(new DefaultTableModel(

newModel,

attributes

));

}

private void getStudents(){

DocumentBuilderFactory dbf=DocumentBuilderFactory.newInstance();

try{

db=dbf.newDocumentBuilder();

try {

document=db.parse("./test.xml");

NodeList s=document.getElementsByTagName("student");

this.model=new String[s.getLength()][attributes.length];

students=new Student[s.getLength()];

int m=0,n=0;

int min=0;

for(int j=0;j<s.getLength();j++){

String value[]=new String[attributes.length];

if(s.item(j).getNodeType()!=Node.TEXT\_NODE){

NodeList temp=s.item(j).getChildNodes();

for(int t=0;t<s.item(j).getChildNodes().getLength();t++){

if(temp.item(t).getNodeType()!=Node.TEXT\_NODE){

value[n]=temp.item(t).getFirstChild().getNodeValue();

//System.out.println(temp.item(t).getFirstChild().getNodeValue());

n++;

}

}

model[m]=value;

students[m]=new Student(value);

m++;

n=0;

}

}

} catch (SAXException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}catch(ParserConfigurationException e ){

}

updateView(model);

}

private Department departments[];

private Student[] students;

private int selectRow=-1;

private String[][] model;

private Document document;

private DocumentBuilder db;

private ArrayList <String> jtreeModel=new ArrayList(4);

private String[] majors={"计算机科学","网络技术","电子信息技术","信息安全"};

private String[] attributes={"学号", "姓名", "性别", "年龄", "出生", "手机", "住址", "邮箱", "专业"};

private String[] attributesXML={"id", "name", "sex", "age", "birthDate", "tel", "address", "email", "major"};

private JTextField findJt;

private JTree jtree;

private JTable table;

private JDialog jd;

private JScrollPane scrollPane;

private JTextField[] jts=new JTextField[attributes.length];

}

添加前：



图3-1 添加前界面

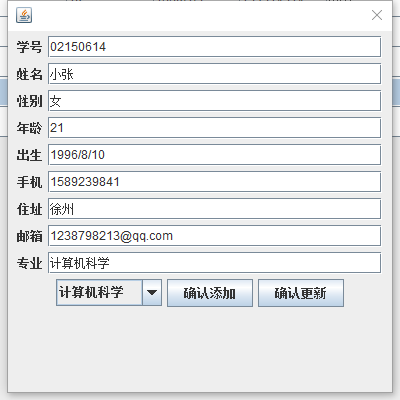


图3-2 编辑界面

添加后：



图3-3 添加后界面

查找前：



图3-4 查找前界面

查找后：

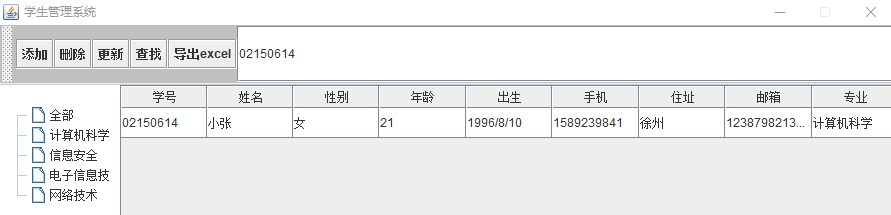


图3-5 查找后界面

分类查看：

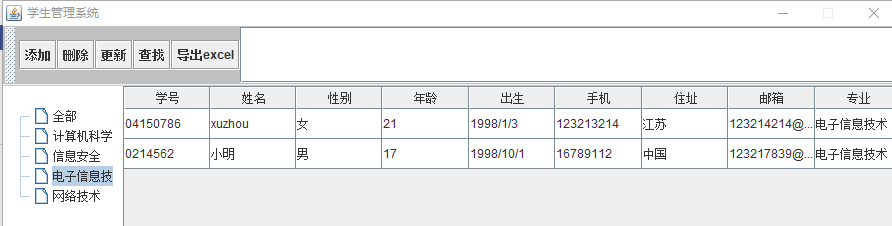


图3-6 分类查看

导出的excel和xml:

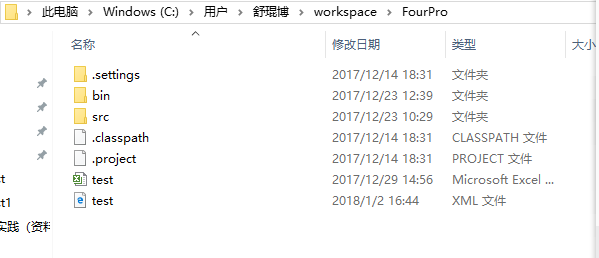


图3-7 导出excel后的文件夹



图3-8 excel文件内容



图3-8 xml文件内容

**4 系统扩展**

在原有项目的基础上扩展了分类功能，可以实现学生信息按专业类别进行管理，便于分类查找，并且增加了导出excel格式文件的功能。

**5 总结**

通过该项目的开发，我了解并学习了xml的基础知识，并掌握了操作xml文件的基本方法，回顾了dom的操作方法，并且对于更删改查的管理系统开发知识有了初步的掌握。

**实验三 拼图游戏**

**1 系统概述**

本项目是一个基于HTML5的在线拼图游戏，同时支持移动端和PC端使用，PC端支持鼠标拖动拼图，移动端支持点击拼图，软件能够自动对内置的图片库中的图片进行分割，并打乱顺序后随机放置在不同的图片框中，开始拼图后，计时功能开启，提供原图查看功能方便用户查看，用户拖动鼠标或点击图片框中的图片进行拼图，在规定时间内拼图成功后系统会自动进行提示。

另外软件支持不同的挑战模式，简单、一般、困难和地狱模式，还提供了用户游戏记录功能，记录每种模式下用户的最高记录，方便下次使用时查看之前的记录。

**2 系统设计**

**2.1 设计目标**

实现图片自动分割功能、图片重排功能、任意图片互换功能、记录游戏的时间、提供原图查看功能、拼图成功检测判断、用户游戏记录功能、能兼容不同的浏览器和设备。

**2.2 设计分析与算法流程**

其主要功能描述如下：

图片动态分割：能自动将图片分割为不同的大小，如3×3矩阵或其它大小的矩阵，便于控制拼图游戏的难易程度。

查看原图：为方便拼图，允许用户在拼图过程中随时查看原图。

随机切换图片：能在软件自身携带的图片中进行随机切换。

鼠标拖动拼图：选择相应图片框中图片后，按下鼠标左键进行拖动，放置到适当位置即可，方便用户操作。

移动端点击拼图：点击选择相应图片框中的图片后，再点击要互换的另一图片进行互换。

自动判断拼图成功：软件自动记忆图片切割后的原始状态，能够对用户拼图是否成功自动做出判断并进行提示。

计时功能：软件能记录游戏的时间，并能实时显示剩余时间。

离线记录功能：软件利用HTML5的localStorage能离线记录用户游戏的最高记录，方便下次游戏时查看各模式的最高记录。

**2.3 界面设计**



图2-3-1 PC浏览器界面



图2-3-2移动端界面

**2.4 关键类图**

**3 系统实现（运行调试）**

本项目的运行环境是浏览器，开发语言使用HTML5、CSS3和Javascript。

主要代码如下：

function createBox(id,side,rightX,rightY){//box对象的工厂函数,box对象包含自身ID,分割图片的信息,分割的大小、位置以及所在图片框的id。

var o=new Object();

o.ID=id;

o.side=side;

o.rightX=rightX;

o.rightY=rightY;

o.posX=rightX;

o.posY=rightY;

o.canvasId=id;

o.getId=function(){

return this.ID;

}

o.getCanvasId=function(){

return this.canvasId;

}

o.setCanvasId=function(newID){

this.canvasId=newID;

}

o.getPositionX=function(){

return this.posX;

}

o.getPositionY=function(){

return this.posY;

}

o.setPostionXY=function(x,y){

this.posX=x;

this.posY=y;

//this.dom.style.left=x+"px";

//this.dom.style.top=y+"px";

}

o.check=function(){//检查图片框的id与box的id是否一致

if(this.canvasId==this.ID){

return true;

}else{

return false;

}

}

return o;

}

function createWrapper(side,posX,posY,dom){

//拼图盘对象的工厂函数,对象包含了图片框数组、边长、位置、以及对应dom

var o=new Object();

o.side=side;

o.posX=posX;

o.canvasArr=[];

o.posY=posY;

o.dom=dom;

o.init=function(){

dom.style.left=this.posX;

dom.style.top=this.posY;

dom.style.width=this.side;

dom.style.height=this.side;

dom.style.position="relative";

}

o.addCanvas=function(canvas){

this.canvasArr.push(canvas);

}

o.swapBox=function(box1,box2,img){//互换两个box的所在图片框id,互换图片框的box的id,通知两个图片框重新绘图

var canvasID2=box2.getCanvasId();

var canvasID1=box1.getCanvasId();

this.canvasArr[canvasID1].draw(box2,img);

box2.setCanvasId(canvasID1);

this.canvasArr[canvasID2].draw(box1,img);

box1.setCanvasId(canvasID2);

this.canvasArr[canvasID1].setDomId(box2.getId());

this.canvasArr[canvasID2].setDomId(box1.getId());

}

o.random=function(boxes,img){

for(var i=0;i<boxes.length;i++){

var r1=Math.floor(Math.random()\*boxes.length);

var r2=Math.floor(Math.random()\*boxes.length);

this.swapBox(boxes[r1],boxes[r2],img);

}

}

o.getPosX=function(){

return this.posX;

}

o.getPosY=function(){

return this.posY;

}

o.getSide=function(){

return this.side;

}

return o;

}

function createCanvas(id,side,posX,posY,dom,divDom){

//图片框数组的工厂函数,图片框的id、边长、位置。Canvas无法绑定点击事件，所以在其上面加一个div,对div绑定事件.

var o=new Object();

o.id=id;

o.side=side;

o.posX=posX;

o.posY=posY;

o.dom=dom;

o.divDom=divDom;

o.setDomId=function(id){

this.divDom.id=id;

}

o.getDomId=function(){

return this.divDom.id;

}

o.init=function(){

dom.width=this.side;

dom.height=this.side;

divDom.id=id;

divDom.style.draggable="true";

divDom.style.left=this.posX+"px";

divDom.style.top=this.posY+"px";

divDom.width=this.side;

divDom.height=this.side;

divDom.style.border="1px blue solid ";

divDom.style.minHeight=this.side+"px";

divDom.style.minWidth=this.side+"px";

divDom.style.position="absolute";

}

o.draw=function(box,img){//绘图函数,根据传入的box的分割图片信息绘图

var context=dom.getContext("2d");

context.drawImage(img,box.getPositionX(),box.getPositionY(),side,side,0,0,side,side);

}

return o;

}

function check(boxes){//检查所有拼图是否放回原位，即box的id与图片框id对应

for(var i=0;i<boxes.length;i++){

if(!boxes[i].check()) return false;

}

return true;

}

function freshDom(){//移除上一局的拼图盘dom,新增一个拼图盘dom

var main=document.createElement("div");

main.id="main";

main.class="main";

document.getElementsByClassName("game")[0].replaceChild(main,document.getElementById("main"));

}

function createRecord(easyDom,normalDom,difficultDom,hardDom){

//记录对象的工厂函数,包含四种记录对应的dom和记录有关的操作

var o=new Object();

o.easyDom=easyDom;

o.normalDom=normalDom;

o.difficultDom=difficultDom;

o.hardDom=hardDom;

o.showRecord=function(){

localStorage.easyRecord?easyDom.firstChild.nodeValue=localStorage.easyRecord:0;

localStorage.normalRecord?normalDom.firstChild.nodeValue=localStorage.normalRecord:0;

localStorage.difficultRecord?difficultDom.firstChild.nodeValue=localStorage.difficultRecord:0;

localStorage.hardRecord?hardDom.firstChild.nodeValue=localStorage.hardRecord:0;

}

o.savaUserRecord=function(newRecord,n){

//根据传入的分割行数存储相应的记录,localStorage在关闭浏览器后有效

if(newRecord!=0){

switch(n){

case 3:

if(!localStorage.easyRecord||newRecord<localStorage.easyRecord){

localStorage.easyRecord=newRecord;

this.showRecord();

//easyDom.firstChild.nodeValue=localStorage.easyRecord;

}

break;

case 4:

if(!localStorage.normalRecord||newRecord<localStorage.normalRecord){

localStorage.normalRecord=newRecord;

this.showRecord();

// normalDom.firstChild.nodeValue=localStorage.normalRecord;

}

break;

case 5:

if(!localStorage.difficultRecord||newRecord<localStorage.difficultRecord){

localStorage.difficultRecord=newRecord;

this.showRecord();

// difficultDom.firstChild.nodeValue=localStorage.difficultRecord;

}

break;

case 8:

if(!localStorage.hardRecord||newRecord<localStorage.hardRecord){

localStorage.hardRecord=newRecord;

this.showRecord();

// hardDom.firstChild.nodeValue=localStorage.hardRecord;

}

break;

}

}

}

return o;

}

function createTime(timeDom,totalTime){

//时间对象的工厂函数，包含计时器和初始时间和剩余时间以及对应dom

var o=new Object();

o.timer=null;

o.timeDom=timeDom;

o.initTime=totalTime;

o.totalTime=totalTime;

o.updateClock=function(){//剩余时间大于零时,将剩余时间-1,并更新显示dom,时间为零时清除计时器

if(totalTime>0){

totalTime-=1;

timeDom.firstChild.nodeValue=totalTime;

}else{

clearInterval(timer);

}

}

o.startTime=function(time){//开始计时函数,设置1s执行一次updateClock函数

clearInterval(timer);

//totalTime=60;

totalTime=time;

initTime=time;

// timeDom=document.getElementById("timer");

timer=setInterval(this.updateClock,1000);

}

o.getTotalTime=function(){

return totalTime;

}

o.getInitTime=function(){

return this.initTime;

}

return o;

}

function createGame(timeObject,recorder){

//Game对象的工厂函数，主要包含拼图盘对象、图片对象、box对象数组、计时对象和记录对象

var o=new Object();

o.currentBox=null;

o.img = null;

o.wrapper=null;

o.boxes=null;

o.timeObject=timeObject;

o.recorder=recorder;

o.key=4;

o.changeImg=function(){

for(var i=0;i<imgs.length;i++){

var r=Math.floor(Math.random()\*imgs.length);

img.src=imgs[r];

}

}

o.startTime=function(totalTime){

timeObject.startTime(totalTime);

}

o.init=function(n){

key=n;

var img = document.getElementById("img");

var wrapDom=document.getElementById("main");

var wrapSide=580;

wrapper=new createWrapper(wrapSide,0,0,wrapDom);

wrapper.init();

var side=wrapSide/n;

boxes=[];

var k=0;

for(var i=0;i<n;i++){//计算分割的图片的边长和位置,动态创建图片框对象和box对象

var posY=side\*i;

for(var j=0;j<n;j++){

var posX=side\*j;

var divDom = document.createElement('div');

var canvasDom = document.createElement('canvas');

var canvas=createCanvas(k,side,posX,posY,canvasDom,divDom);

canvas.init();

wrapper.addCanvas(canvas);

this.addDivAction(divDom);

var box=createBox(k,side,posX,posY);

k++;

boxes.push(box);

canvas.draw(box,img);

wrapDom.appendChild(divDom);

wrapDom.appendChild(canvasDom);

}

}

wrapper.random(boxes,img);

}

o.addDivAction=function(divDom){//为图片框上对应的div绑定各种事件

EventUtil.addHandler(divDom,"touchstart",function(event){

if(sign==false){

currentBox=event.target.id;

sign=true;

}

});

EventUtil.addHandler(divDom,"touchend",function(event){

if(currentBox!=event.target.id&&sign==true){

wrapper.swapBox(boxes[currentBox],boxes[event.target.id],img);

var totalTime=timeObject.getTotalTime();

var initTime=timeObject.getInitTime();

if(totalTime>0&&check(boxes)){

recorder.savaUserRecord(initTime-totalTime,key);

alert("挑战成功！");

}else if(totalTime==0){

alert("超时,挑战失败！");

}

sign=false;

}

});

EventUtil.addHandler(divDom,"dragstart",function(event){

currentBox=event.target.id;

});

EventUtil.addHandler(divDom,"dragover",function(event){

EventUtil.preventDefault(event);

});

EventUtil.addHandler(divDom,"dragenter",function(event){

EventUtil.preventDefault(event);

});

EventUtil.addHandler(divDom,"drop",function(event){

EventUtil.preventDefault(event);

wrapper.swapBox(boxes[currentBox],boxes[event.target.id],img);

var totalTime=timeObject.getTotalTime();

var initTime=timeObject.getInitTime();

if(check(boxes)&&totalTime>0){

recorder.savaUserRecord(initTime-totalTime,key);

alert("挑战成功！");

}else if(totalTime==0){

alert("超时,挑战失败！");

}

});

}

return o;

}

function createButtons(buttonDiv,game){

var o=new Object();

o.buttonDiv=buttonDiv;

o.game=game;

o.clickHandler=function(event){

switch(event.target.id){

case "easy":

freshDom();

game.init(3);

break;

case "normal":

freshDom();

game.init(4);

break;

case "difficult":

freshDom();

game.init(5);

break;

case "hard":

freshDom();

game.init(8);

break;

case "random":

wrapper.random(boxes,img);

break;

case "change":

game.changeImg();

break;

case "start":

game.startTime(60);

wrapper.random(boxes,img);

break;

};

}

o.init=function(){

EventUtil.addHandler(buttonDiv,"click",this.clickHandler)

}

return o;

}

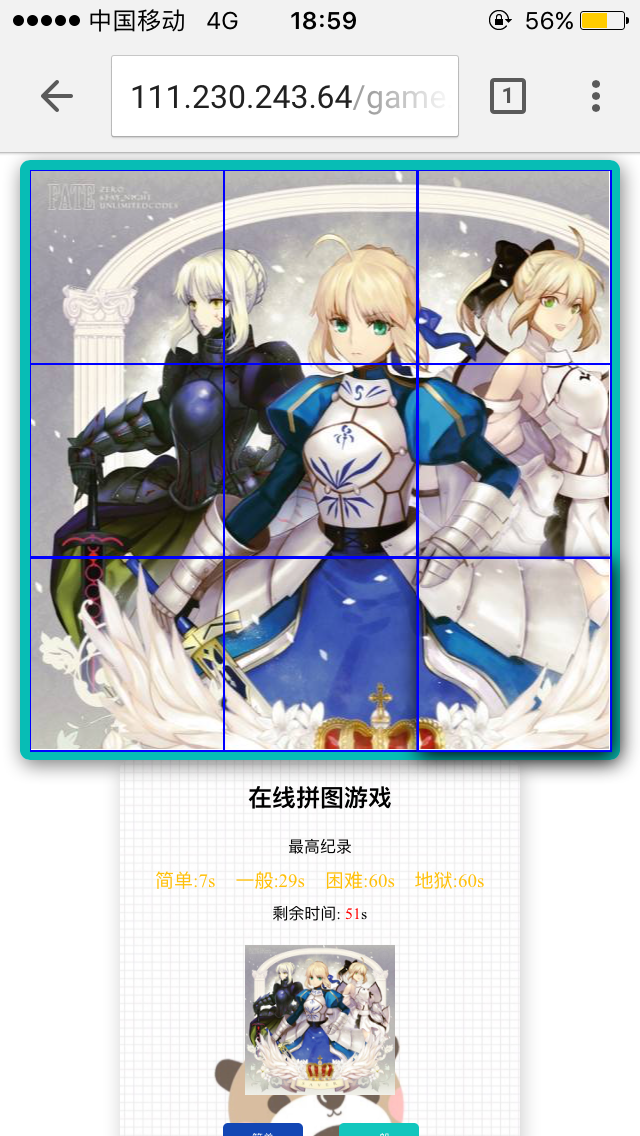


图3-1 挑战成功 图3-2 简单记录更新后

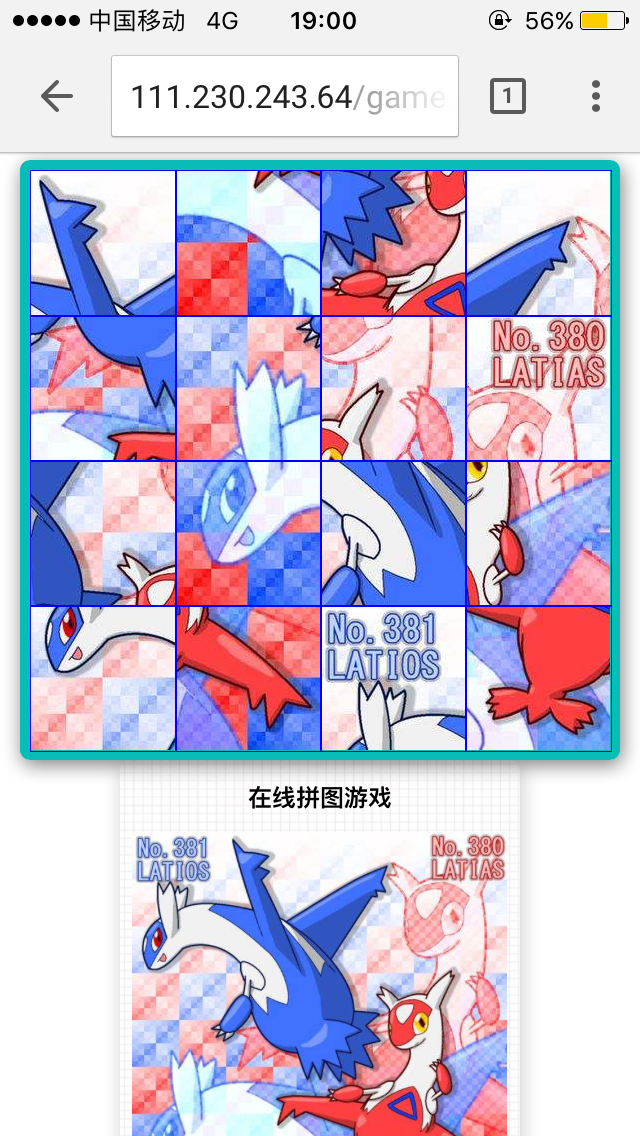
****

图3-1 查看大图

**4 系统扩展**

扩展了模式选择功能，用户可以选择不同难度的模式进行游戏，难度越高，图片划分个数越多，增加了计时功能和离线记录存储功能，可以实时显示游戏剩余时间和离线存储用户的游戏最高记录。

**5 总结**

本项目使用了HTML5和CSS3、JavaScript等前端技术，因为是一个web项目，所以PC浏览器和手机浏览器都能访问，在项目开发过程中，我一开始便尝试使用了面向对象的思想进行编程，设计了拼图盘类、碎片类、图片框类、时间类、记录类、事件类等类，虽然不是使用的Java等面向对象语言，而是使用了Javascript这一基于原型的语言，但语言面向对象的思想都是一致的，在设计过程中主要从软件工程的角度思考了模块与模块之间的关系，对象的属性和方法。尽量减少模块与模块之间的耦合，提高模块的内聚，方便以后扩展，本实验带给我的最大的感受就是面向对象思想的优点，即效率高、易维护，易扩展。

项目在线演示地址: http://111.230.243.64/game.html

**实验四 多文档编辑器**

**1 系统概述**

**2 系统设计**

**2.1 设计目标**

**2.2 设计分析与算法流程**

**2.3 界面设计**

**2.4 关键类图**

**3 系统实现（运行调试）**

**4 系统扩展**

**5 总结**