

期中專題

陳韋翰

eeit_4914號

資料表來源

<div>信筒(箱)設置地點</div> <div>中華郵政公司郵務業務相關資訊</div> <div>評分此資料集：<div><div></div><div></div><div></div><div></div><div></div></div><div>平均 0.00 (0 人次投票)</div></div>	
<div><div><div><div></div><div>瀏覽次數: 9565</div></div><div><div></div><div>下載次數: 924</div></div><div><div></div><div>意見數: 6</div></div></div><div>列印</div></div>	
主要欄位說明	類別代號、類別、信筒樣式代號、信筒樣式、縣市、鄉鎮市區、村里、路名、地址描述、服務單位、聯絡電話、備註、x座標、y座標
*粗體欄位為資料標準欄位	
資料資源下載網址	<div><div><div></div><div>其他</div></div><div><div></div><div>檢視資料</div></div>類別代號(1=限時信筒、2=限時信箱、3=普通信筒、4=普通信箱、5=限時普通雙用信筒、6=限時普通...</div>
提供機關	中華郵政股份有限公司
提供機關聯絡人姓名	中華郵政公司客服中心 女士 (0800-700-365)
更新頻率	不定期
授權方式	政府資料開放授權條款-第1版
計費方式	免費
上架日期	2017-03-09
資料集類型	系統介接程式
詮釋資料更新時間	2021-06-28 16:49
主題分類	其他
服務分類	交通及通訊
資料集分類	開放資料
關鍵字	["信筒(箱)設置地點"]

程式結構

- TitleWindow --> 標題介面
- MainWindow --> 主要使用者介面與DAO呼叫端
- Member --> 資料容器
- Dao --> 資料庫訪問操作
- InsertTableModel --> JTable組成表格模型
- MainTableModel --> JTable組成表格模型

使用資料表內容

類別代號,類別,信筒樣式代號,信筒樣式,縣市,鄉鎮市區,村里,路名,地址描述,服務單位,聯絡電話,備註,x座標,y座標

4,普通信箱,1,單口,雲林縣,四湖鄉,蔡厝村,蔡厝路90號,南光國小前,四湖郵局,(05)7872045,收信時間 11:00,120.219274,23.6121040000000002

4,普通信箱,1,單口,新北市,瑞芳區,上天里,大寮路50號,,瑞芳郵局,(02)24972158#12,收信時間11:00,121.785519,25.102861

4,普通信箱,2,雙口,臺中市,太平區,光華里,大興路345號,,太平宜欣郵局郵務股,(04)23982146#25,收信時間10:20,120.73735,24.15715

3,普通信箱,1,單口,高雄市,路竹區,北嶺里,路科5路100號,守衛室前,路竹郵局,(07)6962311#16,收信時間10:00,120.254936,22.842171999999999

4,普通信箱,1,單口,宜蘭縣,冬山鄉,清溝村,義成路三段168號,台電巷口,羅東郵局郵務股,(03)9561933#301,收信時間10:30,121.765528999999999,24.659393

4,普通信箱,1,單口,彰化縣,溪州鄉,柑園村,下柑路6號,柑園商店,溪州郵局,(04)8895039,收信時間09:00,120.539138,23.822765

4,普通信箱,1,單口,臺東縣,臺東市,建農里,知本路2段92號,民宅前,台東知本郵局,(089)512543,收信時間9:30,121.076700999999999,22.7145410000000004

5,限時普通雙用信箱,2,雙口,苗栗縣,苑裡鎮,社苓里,社苓90之5號,苑裡社苓郵局前,苑裡社苓郵局,(037)741142,收信時間16:50,120.676208,24.404907

4,普通信箱,1,單口,苗栗縣,泰安鄉,中興村,長橋6號,中興代辦所,大湖郵局,(037)992841,收信時間10:00,121.082679,24.404779

3,普通信箱,2,雙口,苗栗縣,銅鑼鄉,九湖村,銅科一路3號,,銅鑼郵局,(037)981133,收信時間12:30,120.76543025502268,24.465697359636728

3,普通信箱,2,雙口,苗栗縣,頭份市,斗煥里,斗煥199之1號,陸軍斗煥營區內,頭份郵局,(037)673114#18,收信時間16:15,120.947036,24.677899

3,普通信箱,2,雙口,高雄市,小港區,店鎮里,中鋼路3號,高雄台船郵局門口,高雄台船郵局,(07)8021442,收信時間16:30,120.356666,22.561111

1,限時信箱,2,雙口,高雄市,楠梓區,中陽里,楠梓新路247號,楠梓郵局前,楠梓郵局,(07)3517250,收信時間17:00,120.325787,22.728381

3,普通信箱,2,雙口,新竹市,香山區,東香里,五福路2段707號,,新竹郵局投遞股,(03)5240075,收信時間12:00,120.952531,24.756901999999999

3,普通信箱,2,雙口,新竹市,香山區,牛埔里,牛埔東路247號,東華街口,新竹郵局投遞股,(03)5240075,收信時間11:00,120.944217,24.796131

4,普通信箱,2,雙口,新竹市,香山區,頂福里,中山路650號,,新竹郵局投遞股,(03)5240075,收信時間10:30,120.954208,24.7957

7,限時信箱,普通信箱,8,普通雙口/限時雙口,新竹縣,芎林鄉,新鳳村,文衡路426號,芎林郵局門口,芎林郵局,(03)5922849,收件時間16:50,121.08377,24.77222

3,普通信箱,2,雙口,高雄市,小港區,松山里,松和路1號,餐旅大學大門,小港投遞股,(07)8064620,收信時間15:30,120.371523,22.56577

1,限時信箱,2,雙口,高雄市,楠梓區,享平里,楠梓路37號,,高北快包股,(07)3109620,收信時間13:00,120.329534,22.728505

1,限時信箱,2,雙口,高雄市,仁武區,竹後里,水管路100-1號,仁武台塑郵局前,仁武台塑郵局,(07)3722811,收信時間17:00,120.336314,22.705947

3,普通信箱,2,雙口,新竹市,北區,金竹里,金竹路99號,,新竹郵局投遞股,(03)5240075,收信時間09:00,120.974002000000001,24.822985

資料庫建置

```
create table mailBoxTable(  
  id int primary key,  
  classNumber int,  
  className varchar(max),  
  classOfMailbox int,  
  mailboxName varchar(max),  
  address varchar(max),  
  descripAddress varchar(max),  
  mechanism varchar(max),  
  phoneNumber Varchar(max),  
  remark varchar(max),  
  longitude float,  
  latitude float  
);
```

```
public class Member {  
    private int id;  
  
    private int classNumber;  
  
    private String className;  
  
    private int classOfMailbox;  
  
    private String mailboxName;  
  
    private String address;  
  
    private String descripAddress;  
  
    private String mechanism;  
  
    private String phoneNumber;  
  
    private String remark;  
  
    private float longitude;  
  
    private float latitude;
```

資料行名稱	資料類型	允許 Null
id	int	<input type="checkbox"/>
classNumber	int	<input checked="" type="checkbox"/>
className	varchar(MAX)	<input checked="" type="checkbox"/>
classOfMailbox	int	<input checked="" type="checkbox"/>
mailboxName	varchar(MAX)	<input checked="" type="checkbox"/>
address	varchar(MAX)	<input checked="" type="checkbox"/>
descripAddress	varchar(MAX)	<input checked="" type="checkbox"/>
mechanism	varchar(MAX)	<input checked="" type="checkbox"/>
phoneNumber	varchar(MAX)	<input checked="" type="checkbox"/>
remark	varchar(MAX)	<input checked="" type="checkbox"/>
longitude	float	<input checked="" type="checkbox"/>
latitude	float	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

變數宣告

```
package mailBoxProject;

import java.awt.EventQueue;

public class MainWindow {

    private JFrame frame;
    private String selectType = "id";
    private String keyword ;
    private static JTable table;
    JScrollPane searchScroll;
    JScrollPane addDataScroll;
    public Vector<String> header = new Vector<String>();
    private JTextField keywordEnter;
    private JButton insertButton;
    private JTable addDataTable;
    private Vector<Vector<String>> VectorBig = new Vector<Vector<String>>();
    private Vector<String> vectorSmall = new Vector<String>();
    MainTableModel model;
    private JTextPane userGuide;
    private String path;
    private JButton exportButton;
```

功能展示與說明—UI



UI—根據檔案形式自動匯入

```
//透過選擇檔案匯入資料
JButton importButton = new JButton("上傳檔案");
importButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
importButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
importButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent clicked) {
        int result = 0;
        JFileChooser fileChooser = new JFileChooser();
        FileSystemView fsv = FileSystemView.getFileSystemView();
        System.out.println(fsv.getHomeDirectory());
        fileChooser.setCurrentDirectory(fsv.getHomeDirectory());
        fileChooser.setDialogTitle("選擇檔案");
        fileChooser.setApproveButtonText("確定");
        fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
        result = fileChooser.showOpenDialog(null);
        if (JFileChooser.APPROVE_OPTION == result) {
            path = fileChooser.getSelectedFile().getPath();
            System.out.println(path);
            userGuide.setText(path);
            int index = path.lastIndexOf(".");
            String filter = path.substring(index+1);
            switch (filter) {
                case "csv":
                    try {
                        int rows = dao.insertDataByCSV(new File(path));
                        String text = "完成，總共新增 " + rows + " 筆資料，請重新點擊查詢";
                        userGuide.setText(text);
                    } catch (SQLException | IOException e) {
                        e.printStackTrace();
                    }
                    break;
                case "json":
                    try {
                        int rows = dao.insertDataByJson(new File(path));
                        String text = "完成，總共新增 " + rows + " 筆資料，請重新點擊查詢";
                        userGuide.setText(text);
                    } catch (SQLException | IOException e) {
                        e.printStackTrace();
                    }
                    break;
                case "xml":
                    try {
                        int rows = dao.insertDataByXML(new File(path));
                        String text = "完成，總共新增 " + rows + " 筆資料，請重新點擊查詢";
                        userGuide.setText(text);
                    } catch (SAXException | IOException | SQLException e) {
                        e.printStackTrace();
                    }
                    break;
                default:
                    userGuide.setText("檔案類型錯誤!!");
                    break;
            }
        }
    }
});
```


匯入CSV、JSON資料

```
//匯入CSV資料
public int insertDataByCSV(File file) throws SQLException, IOException {
    try (FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);
        ) {
        this.createConnection();
        String sqlInsert = "insert into mailBoxTable values(?,?,?,?,?,?,?,?,?,?,?)";
        String sqlGetLastId = "select top 1 id from mailBoxTable order by id desc";
        PreparedStatement preStateId = conn.prepareStatement(sqlGetLastId);
        ResultSet rsId = preStateId.executeQuery();
        int id = 0;
        if(rsId.next()) {
            id = rsId.getInt(1);
        }
        idStart = id;
        int times = 0;
        rsId.close();
        preStateId.close();
        PreparedStatement preStateInsert = conn.prepareStatement(sqlInsert);
        br.readLine();
        String line = null;
        while((line = br.readLine()) != null) {
            id++;
            String[] splitline = line.split(",");
            Member m = new Member();
            m.setId(id);
            m.setClassNumber(Integer.parseInt(splitline[0]));
            m.setClassName(splitline[1]);
            m.setClassOfMailbox(Integer.parseInt(splitline[2]));
            m.setMailboxName(splitline[3]);
            m.setAddress(splitline[4]+splitline[5]+splitline[6]+splitline[7]);
            m.setDescripAddress(splitline[8]);
            m.setMechanism(splitline[9]);
            m.setPhoneNumber(splitline[10]);
            m.setRemark(splitline[11]);
            m.setLongitude(Float.parseFloat(splitline[12]));
            m.setLatitude(Float.parseFloat(splitline[13]));
            preStateInsert.setInt(1, m.getId());
            preStateInsert.setInt(2, m.getClassNumber());
            preStateInsert.setString(3, m.getClassName());
            preStateInsert.setInt(4, m.getClassOfMailbox());
            preStateInsert.setString(5, m.getMailboxName());
            preStateInsert.setString(6, m.getAddress());
            preStateInsert.setString(7, m.getDescripAddress());
            preStateInsert.setString(8, m.getMechanism());
            preStateInsert.setString(9, m.getPhoneNumber());
            preStateInsert.setString(10, m.getRemark());
            preStateInsert.setFloat(11, m.getLongitude());
            preStateInsert.setFloat(12, m.getLatitude());
            preStateInsert.executeUpdate();
            times++;
        }
        preStateInsert.close();
        System.out.println("完成，共新增" + times + "筆資料");
        idEnd = idStart + times;
        return times;
    } finally {
        this.closeConnection();
    }
}
```

```
//匯入JSON資料
public int insertDataByJson(File file) throws SQLException, IOException {
    int times = 0;
    try (FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);
        ) {
        this.createConnection();
        String sql = "insert into mailBoxTable values(?,?,?,?,?,?,?,?,?,?,?)";
        String sqlGetLastId = "select top 1 id from mailBoxTable order by id desc";
        PreparedStatement preStateId = conn.prepareStatement(sqlGetLastId);
        ResultSet rsId = preStateId.executeQuery();
        int id = 0;
        if(rsId.next()) {
            id = rsId.getInt(1);
        }
        idStart = id;
        rsId.close();
        preStateId.close();
        PreparedStatement preState = conn.prepareStatement(sql);
        JsonElement parseReader = JsonParser.parseReader(br);
        JSONArray JArray = parseReader.getAsJsonArray();
        for(JsonElement jelement:JArray) {
            id++;
            JsonObject jobject = jelement.getAsJsonObject();
            int classNumber = jobject.get("類別代號").getAsInt();
            String className = jobject.get("類別").getString();
            int classOfMailbox = jobject.get("信箱樣式代號").getAsInt();
            String mailboxName = jobject.get("信箱樣式").getString();
            String address1 = jobject.get("縣市").getString();
            String address2 = jobject.get("鄉鎮市區").getString();
            String address3 = jobject.get("村里").getString();
            String address4 = jobject.get("路名").getString();
            String descripAddress = jobject.get("地址描述").getString();
            String mechanism = jobject.get("服務單位").getString();
            String phoneNumber = jobject.get("聯絡電話").getString();
            String remark = jobject.get("備註").getString();
            float longitude = jobject.get("x座標").getAsFloat();
            float latitude = jobject.get("y座標").getAsFloat();
            times++;
            Member m = new Member(classNumber, className, classOfMailbox, mailboxName,
                address1+address2+address3+address4, descripAddress, mechanism,
                phoneNumber, remark, longitude, latitude);
            m.setId(id);
            preState.setInt(1, m.getId());
            preState.setInt(2, m.getClassNumber());
            preState.setString(3, m.getClassName());
            preState.setInt(4, m.getClassOfMailbox());
            preState.setString(5, m.getMailboxName());
            preState.setString(6, m.getAddress());
            preState.setString(7, m.getDescripAddress());
            preState.setString(8, m.getMechanism());
            preState.setString(9, m.getPhoneNumber());
            preState.setString(10, m.getRemark());
            preState.setFloat(11, m.getLongitude());
            preState.setFloat(12, m.getLatitude());
            preState.executeUpdate();
        }
        preState.close();
        System.out.println("完成，共新增" + times + "筆資料");
        idEnd = idStart + times;
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } finally {
        this.closeConnection();
    }
    return times;
}
```

匯入XML資料

```
//匯入XML資料
public int insertDataByXML(File file) throws SAXException, IOException, SQLException {
    int times = 0;
    try(FileReader fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);
        ) {
        SAXReader reader = new SAXReader();
        Document document = reader.read(br);

        List<Node> nodes = document.selectNodes("table/row");
        this.createConnection();
        String sqlInsert = "insert into mailBoxTable values(?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
        String sqlGetLastId = "select top 1 id from mailBoxTable order by id desc";
        PreparedStatement preStateId = conn.prepareStatement(sqlGetLastId);
        ResultSet rsId = preStateId.executeQuery();
        int id = 0;
        if(rsId.next()) {
            id = rsId.getInt(1);
        }
        idStart = id;
        rsId.close();
        preStateId.close();
        PreparedStatement preStateInsert = conn.prepareStatement(sqlInsert);
        for(Node node : nodes) {
            id++;
            int classNumber = Integer.parseInt(node.selectSingleNode("Col1").getText());
            String className = node.selectSingleNode("Col2").getText();
            int classofMailBox = Integer.parseInt(node.selectSingleNode("Col3").getText());
            String mailboxName = node.selectSingleNode("Col4").getText();
            String address1 = node.selectSingleNode("Col5").getText();
            String address2 = node.selectSingleNode("Col6").getText();
            String address3 = node.selectSingleNode("Col7").getText();
            String address4 = node.selectSingleNode("Col8").getText();
            String descripAddress = node.selectSingleNode("Col9").getText();
            String mechanism = node.selectSingleNode("Col10").getText();
            String phoneNumber = node.selectSingleNode("Col11").getText();
            String remark = node.selectSingleNode("Col12").getText();
            float longitude = Float.valueOf(node.selectSingleNode("Col13").getText());
            float latitude = Float.valueOf(node.selectSingleNode("Col14").getText());
            Member m = new Member(classNumber, className, classofMailBox, mailboxName, address1+address2+address3+address4, descripAddress, mechanism, phoneNumber, remark, longitude, latitude);
            times++;
            m.setId(id);
            preStateInsert.setInt(1, m.getId());
            preStateInsert.setInt(2, m.getClassNumber());
            preStateInsert.setString(3, m.getClassName());
            preStateInsert.setInt(4, m.getClassofMailBox());
            preStateInsert.setString(5, m.getMailboxName());
            preStateInsert.setString(6, m.getAddress());
            preStateInsert.setString(7, m.getDescripAddress());
            preStateInsert.setString(8, m.getMechanism());
            preStateInsert.setString(9, m.getPhoneNumber());
            preStateInsert.setString(10, m.getRemark());
            preStateInsert.setFloat(11, m.getLongitude());
            preStateInsert.setFloat(12, m.getLatitude());
            preStateInsert.executeUpdate();
        }
        preStateInsert.close();
        System.out.println("完成，共新增" + times + "筆資料");
        idEnd = idStart + times;
    } catch (Exception e) {
        e.printStackTrace();
    } finally{
        this.closeConnection();
    }
    return times;
}
```

查詢資料

```
//可選擇以何種資料進行查詢的comboBox
JComboBox<String> typeCombo = new JComboBox<String>();
typeCombo.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
typeCombo.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
typeCombo.addItemListener(new ItemListener() {
    public void itemStateChanged(ItemEvent e) {
        if (e.getStateChange() == ItemEvent.SELECTED) {
            switch ((String)e.getItem()) {
                case "ID":
                    selectType = "id";
                    break;
                case "類別編號":
                    selectType = "classNumber";
                    break;
                case "類別":
                    selectType = "className";
                    break;
                case "信筒樣式代號":
                    selectType = "classOfMailbox";
                    break;
                case "信筒樣式":
                    selectType = "mailboxName";
                    break;
                case "地址":
                    selectType = "address";
                    break;
                case "地址描述":
                    selectType = "descripAddress";
                    break;
                case "服務單位":
                    selectType = "mechanism";
                    break;
                case "聯絡電話":
                    selectType = "phoneNumber";
                    break;
                case "備註":
                    selectType = "remark";
                    break;
                case "x座標":
                    selectType = "longitude";
                    break;
                case "y座標":
                    selectType = "latitude";
                    break;
                default:
                    selectType = "?";
                    break;
            }
        }
    }
});
typeCombo.setModel(new DefaultComboBoxModel(new String[] {"ID", "類別編號",
    "類別", "信筒樣式代號", "信筒樣式", "地址", "地址描述", "服務單位",
    "聯絡電話", "備註", "x座標", "y座標"}));
```

```
//依輸入內容查詢資料
keywordEnter = new JTextField();
keywordEnter.setColumns(10);

JButton searchButton = new JButton("查詢");
searchButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
searchButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
searchButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        keyword = "% " + keywordEnter.getText() + "% ";
        try {
            model = new MainTableModel(dao.searchByKey(selectType, keyword), header);
            table = new JTable(model);
            searchScroll.setViewportView(table);
            table.setPreferredScrollableViewportSize(frame.getSize());
            userGuide.setText("可直接點擊表格內容進行修改，或選取欄位後點擊左方按鈕刪除整行檔案");
        } catch (SQLException e) {
            e.printStackTrace();
        }
        table.getColumnModel().getColumn(0).setPreferredWidth(50);
        table.getColumnModel().getColumn(1).setPreferredWidth(20);
        table.getColumnModel().getColumn(3).setPreferredWidth(20);
        table.getColumnModel().getColumn(4).setPreferredWidth(30);
        table.getColumnModel().getColumn(5).setPreferredWidth(200);
        table.getColumnModel().getColumn(8).setPreferredWidth(130);
        table.getColumnModel().getColumn(9).setPreferredWidth(110);
    }
});
```

```
//依照關鍵字查找資料
public Vector<Vector<String>> searchByKey(String selectType, String keyword) throws SQLException {
    Vector<Vector<String>> vecBig = new Vector<Vector<String>>();
    try {
        String sql = "select * from mailBoxTable ";
        if (selectType != null) {
            sql += " where " + selectType + " like " + keyword;
        }
        this.createConnection();
        PreparedStatement preState = conn.prepareStatement(sql);
        ResultSet rs = preState.executeQuery();

        while(rs.next()) {
            Vector<String> vecSmall = new Vector<String>();
            vecSmall.add(rs.getString(1));
            vecSmall.add(rs.getString(2));
            vecSmall.add(rs.getString(3));
            vecSmall.add(rs.getString(4));
            vecSmall.add(rs.getString(5));
            vecSmall.add(rs.getString(6));
            vecSmall.add(rs.getString(7));
            vecSmall.add(rs.getString(8));
            vecSmall.add(rs.getString(9));
            vecSmall.add(rs.getString(10));
            vecSmall.add(rs.getString(11));
            vecSmall.add(rs.getString(12));
            vecBig.add(vecSmall);
        }
        rs.close();
        preState.close();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        this.closeConnection();
    }
    return vecBig;
}
```

修改資料

```
//依照ID更改資料
public void updateDataById(String selectType,String newValue,String idValue) throws SQLException {
    switch (selectType) {
        case "類別編號":
            selectType = "classNumber";
            break;
        case "類別":
            selectType = "className";
            break;
        case "信箱樣式代號":
            selectType = "classOfMailbox";
            break;
        case "信箱樣式":
            selectType = "mailboxName";
            break;
        case "地址":
            selectType = "address";
            break;
        case "地址描述":
            selectType = "descripAddress";
            break;
        case "服務單位":
            selectType = "mechanism";
            break;
        case "聯絡電話":
            selectType = "phoneNumber";
            break;
        case "備註":
            selectType = "remark";
            break;
        case "x座標":
            selectType = "longitude";
            break;
        case "y座標":
            selectType = "latitude";
            break;
        default:
            selectType = "?";
            break;
    }
    String sql = "update mailBoxTable set ";
    sql += selectType + " = '"+newValue+"' where id = "+idValue;
    System.out.println(sql);
    this.createConnection();
    try{
        PreparedStatement preState = conn.prepareStatement(sql);

        int row = preState.executeUpdate();

        System.out.println("修改了" + row + "筆資料");
        preState.close();
    }finally {
        this.closeConnection();
    }
}
```

```
public Vector<Vector<String>> searchByKey(String selectType,String keyword) throws SQLException{
    Vector<Vector<String>> vecBig = new Vector<Vector<String>>();
    try {
        String sql = "select * from mailBoxTable ";
        if(selectType != null) {
            sql += " where " + selectType + " like " +keyword;
        }
        this.createConnection();
        PreparedStatement preState = conn.prepareStatement(sql);
        ResultSet rs = preState.executeQuery();

        while(rs.next()) {
            Vector<String> vecSmall = new Vector<String>();
            vecSmall.add(rs.getString(1));
            vecSmall.add(rs.getString(2));
            vecSmall.add(rs.getString(3));
            vecSmall.add(rs.getString(4));
            vecSmall.add(rs.getString(5));
            vecSmall.add(rs.getString(6));
            vecSmall.add(rs.getString(7));
            vecSmall.add(rs.getString(8));
            vecSmall.add(rs.getString(9));
            vecSmall.add(rs.getString(10));
            vecSmall.add(rs.getString(11));
            vecSmall.add(rs.getString(12));
            vecBig.add(vecSmall);
        }
        rs.close();
        preState.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }finally {
        this.closeConnection();
    }
    return vecBig;
}
```

自定義表格MODEL

```
package mailBoxProject;

import java.sql.SQLException;

public class InsertTableModel extends AbstractTableModel{

    Vector<Vector<String>> data;
    Vector<String> columns;

    InsertTableModel(Vector<Vector<String>> data, Vector<String> columns){
        this.data = data;
        this.columns = columns;
    }

    @Override
    public int getRowCount() {
        return data.size();
    }

    @Override
    public int getColumnCount() {
        return columns.size();
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        Object string = data.get(rowIndex).get(columnIndex);
        return string;
    }

    @Override
    public String getColumnName(int column) {
        return columns.get(column);
    }

    @Override
    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return true;
    }

    @Override
    public void setValueAt(Object aValue, int rowIndex, int columnIndex) {
        Vector<String> rowdata = data.get(rowIndex);
        rowdata.set(columnIndex, (String) aValue);
        fireTableCellUpdated(rowIndex, columnIndex);
    }
}
```

```
package mailBoxProject;

import java.sql.SQLException;

public class MainTableModel extends AbstractTableModel{

    Vector<Vector<String>> data;
    Vector<String> columns;

    MainTableModel(Vector<Vector<String>> data, Vector<String> columns){
        this.data = data;
        this.columns = columns;
    }

    public String getColumnrow(int rowIndex) {
        return columns.get(rowIndex);
    }

    @Override
    public int getRowCount() {
        return data.size();
    }

    @Override
    public int getColumnCount() {
        return columns.size();
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        Object string = data.get(rowIndex).get(columnIndex);
        return string;
    }

    @Override
    public String getColumnName(int column) {
        return columns.get(column);
    }

    @Override
    public boolean isCellEditable(int rowIndex, int columnIndex) {
        if(columnIndex==0) {
            return false;
        }else {
            return true;
        }
    }

    @Override
    public void setValueAt(Object aValue, int rowIndex, int columnIndex) {
        try {
            new Dao().updateDataById(this.getColumnName(columnIndex), (String) aValue, (String) this.getValueAt(rowIndex, 0));
        } catch (SQLException e) {
            System.out.println("匯入資料發生問題:");
            e.printStackTrace();
        }
        Vector<String> rowdata = data.get(rowIndex);
        rowdata.set(columnIndex, (String) aValue);
        fireTableCellUpdated(rowIndex, columnIndex);
    }
}
```

新增資料

```
//匯入單筆資料
public void insertDataBy1(Vector<String> data) throws SQLException, IOException {
    this.createConnection();
    String sqlInsert = "insert into mailBoxTable values(?,?,?,?,?,?,?,?,?,?)";
    int times = 0;
    PreparedStatement preStateInsert = conn.prepareStatement(sqlInsert);
    preStateInsert.setString(1, (String)data.get(0));
    preStateInsert.setString(2, (String)data.get(1));
    preStateInsert.setString(3, data.get(2));
    preStateInsert.setString(4, (String)data.get(3));
    preStateInsert.setString(5, data.get(4));
    preStateInsert.setString(6, data.get(5));
    preStateInsert.setString(7, data.get(6));
    preStateInsert.setString(8, data.get(7));
    preStateInsert.setString(9, data.get(8));
    preStateInsert.setString(10, data.get(9));
    preStateInsert.setString(11, (String)data.get(10));
    preStateInsert.setString(12, (String)data.get(11));
    preStateInsert.executeUpdate();
    times++;
    preStateInsert.close();
    System.out.println("完成，共新增" + times + "筆資料");
    idEnd = idStart + times;
}
```

```
//新增單筆資料
insertButton = new JButton("新增資料");
insertButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
insertButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
insertButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            dao.insertDataBy1(vectorSmall);
            VectorBig.clear();
            for(int i = 1 ; i <= 11 ; i++) {
                vectorSmall.set(i, null);
            }
            VectorBig.add(vectorSmall);

            addDataTable.setModel(new InsertTableModel(VectorBig,header));
            addDataScroll.setViewportViewView(addDataTable);
            addDataTable.setPreferredScrollableViewportSize(frame.getSize());

            model = new MainTableModel(dao.searchByKey(selectType, keyword),header);
            table = new JTable(model);
            searchScroll.setViewportViewView(table);
            table.getColumnModel().getColumn(0).setPreferredWidth(50);
            table.getColumnModel().getColumn(1).setPreferredWidth(20);
            table.getColumnModel().getColumn(3).setPreferredWidth(20);
            table.getColumnModel().getColumn(4).setPreferredWidth(30);
            table.getColumnModel().getColumn(5).setPreferredWidth(200);
            table.getColumnModel().getColumn(8).setPreferredWidth(130);
            table.getColumnModel().getColumn(9).setPreferredWidth(110);
            userGuide.setText("新增資料成功，新增ID:" + vectorSmall.get(0));
            vectorSmall.set(0, null);
        } catch (SQLException e1) {
            userGuide.setText("失敗，輸入了錯誤的ID");
            e1.printStackTrace();
        }
        catch (IOException e1) {
            userGuide.setText("發生未知錯誤");
            e1.printStackTrace();
        }
    }
});
```

刪除資料

```
//點擊一行資料後進行刪除
JButton deleteByRowButton = new JButton("整行刪除");
deleteByRowButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
deleteByRowButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
deleteByRowButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            Object rowID = model.getValueAt(table.getSelectedRow(), 0);
            int deleteId = Integer.parseInt((String)rowID);
            dao.deleteById(deleteId);
            model = new MainTableModel(dao.searchByKey(selectType, keyword), header);
            table = new JTable(model);
            searchScroll.setViewportView(table);
            table.getColumnModel().getColumn(0).setPreferredWidth(50);
            table.getColumnModel().getColumn(1).setPreferredWidth(20);
            table.getColumnModel().getColumn(3).setPreferredWidth(20);
            table.getColumnModel().getColumn(4).setPreferredWidth(30);
            table.getColumnModel().getColumn(5).setPreferredWidth(200);
            table.getColumnModel().getColumn(8).setPreferredWidth(130);
            table.getColumnModel().getColumn(9).setPreferredWidth(110);
            userGuide.setText("刪除ID: " + deleteId);
        } catch (SQLException e1) {
            userGuide.setText("資料庫連接失敗!!");
            e1.printStackTrace();
        } catch (ArrayIndexOutOfBoundsException e1) {
            userGuide.setText("選取資料錯誤");
            e1.printStackTrace();
        }
    }
});
```

```
//透過ID刪除資料
public void deleteById(int id) throws SQLException {
    try {
        String sql = "delete from mailBoxTable where Id = ?" ;
        this.createConnection();
        PreparedStatement preState = conn.prepareStatement(sql);
        preState.setInt(1, id);
        int row = preState.executeUpdate();
        preState.close();
        System.out.println("成功，刪除" + row + "筆資料");
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        this.closeConnection();
    }
}
```

匯出查詢結果—CSV

```
//以選擇匯出模式執行匯出
exportButton = new JButton("結果匯出");
exportButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
exportButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
exportButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        int result = 0;
        JFileChooser fileChooser = new JFileChooser();
        FileSystemView fsv = FileSystemView.getFileSystemView();
        fileChooser.setCurrentDirectory(fsv.getHomeDirectory());
        fileChooser.setDialogTitle("選擇匯出位置");
        fileChooser.setApproveButtonText("確定");
        fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        result = fileChooser.showOpenDialog(null);
        if(JFileChooser.APPROVE_OPTION == result) {
            path = fileChooser.getSelectedFile().getPath();
            path = path + "\\export" + new Date(System.currentTimeMillis());
            userGuide.setText("匯出中...");
            System.out.println(exportCombo.getSelectedItem());
            try {
                switch((String)exportCombo.getSelectedItem()) {
                    case "CSV":
                        path += ".csv";
                        dao.exportDataCSV(path, selectType, keyword);
                        userGuide.setText("匯出完成，檔案名稱為: " + path);
                        break;
                    case "JSON":
                        path += ".json";
                        dao.exportDataJson(path, selectType, keyword);
                        userGuide.setText("匯出完成，檔案名稱為: " + path);
                        break;
                    default:
                        userGuide.setText("檔案類型錯誤!!");
                        break;
                }
            } catch (SQLException e1) {
                e1.printStackTrace();
            } catch (IOException e1) {
                e1.printStackTrace();
            }
        }
    }
});
```

```
//匯出csv檔，把加入的id刪除
public void exportDataCSV(String filepath, String selectType, String keyword) throws SQLException, IOException {
    String sqlOutput = "select * from mailBoxTable ";
    this.createConnection();
    if(keyword != null) {
        sqlOutput += " where " + selectType + " like " + keyword;
    }
    System.out.println(sqlOutput);
    PreparedStatement preStateOutput = conn.prepareStatement(sqlOutput);
    File file = new File(filepath);
    try(FileOutputStream fos = new FileOutputStream(file);
        OutputStreamWriter osw = new OutputStreamWriter(fos, "UTF8")) {
        String header = new String("類別代號"+"、" + "類別" + "、" + "信筒樣式代號" + "、" + "信筒樣式" + "、" + "地址" + "、" + "地址描述" + "、" + "服務單位" + "、" + "聯絡電話" + "、" + "備註" + "、" + "x座標" + "、" + "y座標" + "\n");
        System.out.println(header);
        osw.write(header);
        osw.flush();
        ResultSet rs = preStateOutput.executeQuery();
        while(rs.next()) {
            String result = rs.getString(2) + "、" + rs.getString(3) + "、" + rs.getString(4) + "、" + rs.getString(5) + "、" + rs.getString(6) + "、" + rs.getString(7) + "、" + rs.getString(8) + "、" + rs.getString(9) + "、" + rs.getString(10) + "、" + rs.getString(11) + "、" + rs.getString(12) + "\n";
            osw.write(result);
            osw.flush();
        }
        preStateOutput.close();
        this.closeConnection();
    }
}
```


匯出查詢結果—JSON

```
//以選擇匯出模式執行匯出
exportButton = new JButton("結果匯出");
exportButton.setBorder(new LineBorder(new Color(0, 0, 0), 1, true));
exportButton.setFont(new Font("微軟正黑體", Font.PLAIN, 12));
exportButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        int result = 0;
        JFileChooser fileChooser = new JFileChooser();
        FileSystemView fsv = FileSystemView.getFileSystemView();
        fileChooser.setCurrentDirectory(fsv.getHomeDirectory());
        fileChooser.setDialogTitle("選擇匯出位置");
        fileChooser.setApproveButtonText("確定");
        fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        result = fileChooser.showOpenDialog(null);
        if(JFileChooser.APPROVE_OPTION == result) {
            path = fileChooser.getSelectedFile().getPath();
            path = path + "\\export" + new Date(System.currentTimeMillis());
            userGuide.setText("匯出中...");
            System.out.println(exportCombo.getSelectedItem());
            try {
                switch((String)exportCombo.getSelectedItem()) {
                    case "CSV":
                        path += ".csv";
                        dao.exportDataCSV(path, selectType, keyword);
                        userGuide.setText("匯出完成，檔案名稱為: " + path);
                        break;
                    case "JSON":
                        path += ".json";
                        dao.exportDataJson(path, selectType, keyword);
                        userGuide.setText("匯出完成，檔案名稱為: " + path);
                        break;
                    default:
                        userGuide.setText("檔案類型錯誤!!");
                        break;
                }
            } catch (SQLException e1) {
                e1.printStackTrace();
            } catch (IOException e1) {
                e1.printStackTrace();
            }
        }
    }
});
```

```
//匯出json檔案
public void exportDataJson(String filepath, String selectType, String keyword) throws SQLException, IOException {
    String sqlOutput = "select * from mailBoxTable ";
    if(keyword != null) {
        sqlOutput += " where " + selectType + " like " + keyword + " for json path";
    } else {
        sqlOutput += " for json path";
    }
    this.createConnection();
    PreparedStatement preState = conn.prepareStatement(sqlOutput);
    File file = new File(filepath);
    try(FileOutputStream fos = new FileOutputStream(file);
        OutputStreamWriter osw = new OutputStreamWriter(fos, "UTF8");
        BufferedWriter bw = new BufferedWriter(osw);
    ){
        ResultSet rs = preState.executeQuery();
        while(rs.next()) {
            String result = rs.getString(1);
            bw.write(result);
            bw.flush();
        }
        preState.close();
        rs.close();
    } finally {
        this.closeConnection();
    }
}
```

參考資料

政府資料開放平台網頁:信筒(箱)設置地點・信箱位置資料表來源 (2017年3月9日上架)

檢自<https://data.gov.tw/dataset/7964> (2022年7月3日)

熊貓辦公--鎖孔查詢・鎖孔圖片來源(上架時間未知)

檢自<https://www.tukuppt.com/muban/lwwdwagl.html>(2022年7月3日)

維基百科:中華郵政郵徽・標題icon圖片來源(上架時間未知)

檢自https://zh.wikipedia.org/wiki/%E4%B8%AD%E8%8F%AF%E9%83%B5%E6%94%BF#/media/File:Chunghwa_Post_Logo.svg(2022年7月3日)

Gson.jar程式來源(上架時間未知)

檢自<https://github.com/google/gson>(2022年7月3日)

Dom4j・Dom4j程式來源(上架時間未知)

檢自<https://dom4j.github.io/>(2022年7月3日)

eclipse官網・windowbuilder程式來源(上架時間未知)

檢自<https://www.eclipse.org/windowbuilder/>(2022年7月3日)

參考資料

Jaxen · Dom4j補助程式來源(上架時間未知)

檢自<https://github.com/jaxen-xpath/jaxen>(2022年7月3日)

小狐狸事務所-Java Swing 測試 : 表格 JTable(2014年5月24日)

檢自<http://yhhuang1966.blogspot.com/2014/05/java-swing-jtable.html>(2022年7月3日)

Swing簡單的檔案上傳-程式人生(2018-12-02)

檢自<https://www.796t.com/content/1543754104.html>(2022年7月3日)

Java視窗程式開發WindowBuilder (使用教學)(2017-05-28)

檢自<https://xenby.com/b/94-java%E8%A6%96%E7%AA%97%E7%A8%8B%E5%BC%8F%E9%96%8B%E7%99%BCwindowbuilder-%E4%BD%BF%E7%94%A8%E6%95%99%E5%AD%B8>(2022年7月3日)

謝謝您的聆聽！