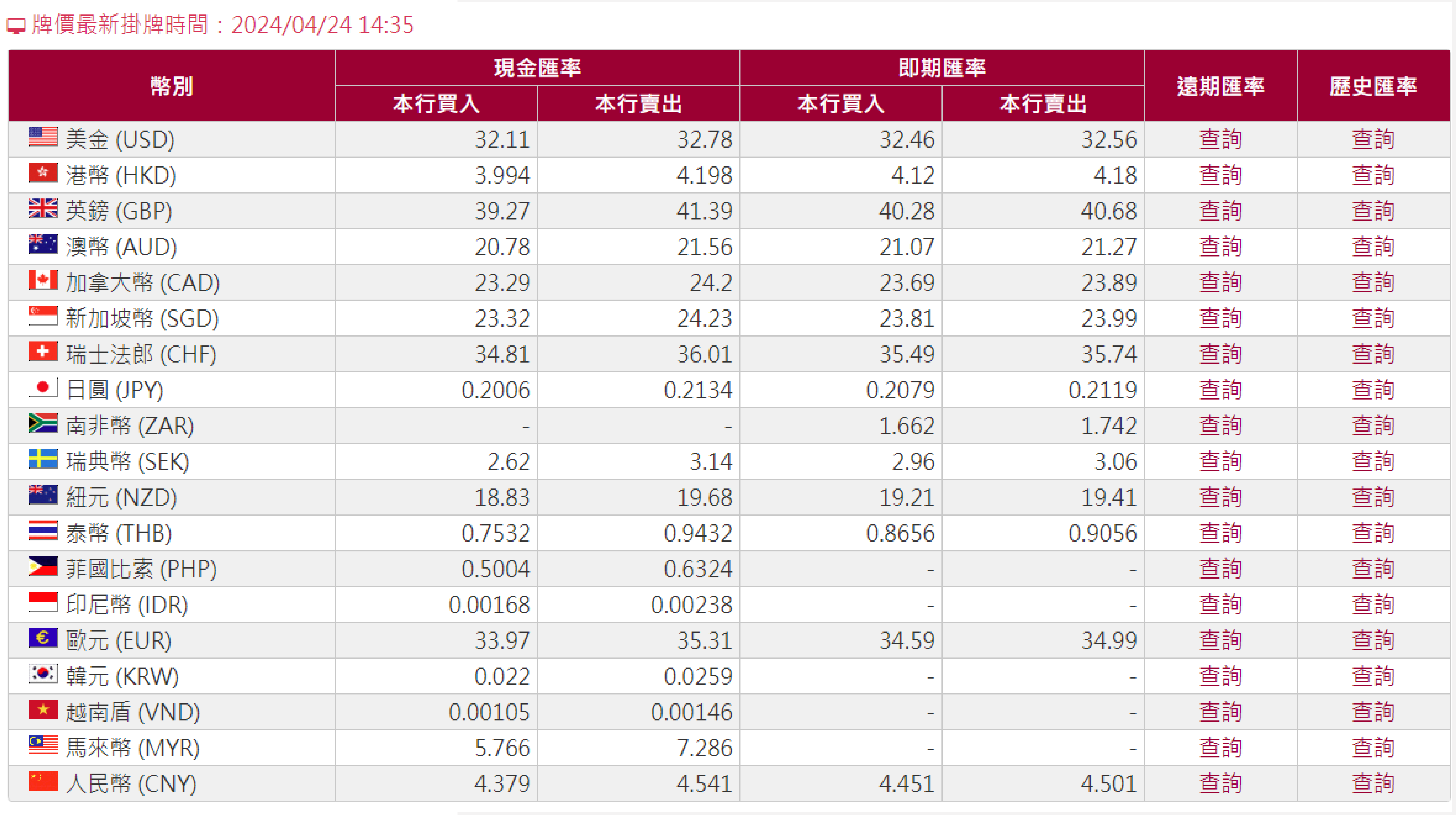
專題報告
匯率資料庫

鄧晴文

26 April, 2024

OVERVIEW

動機



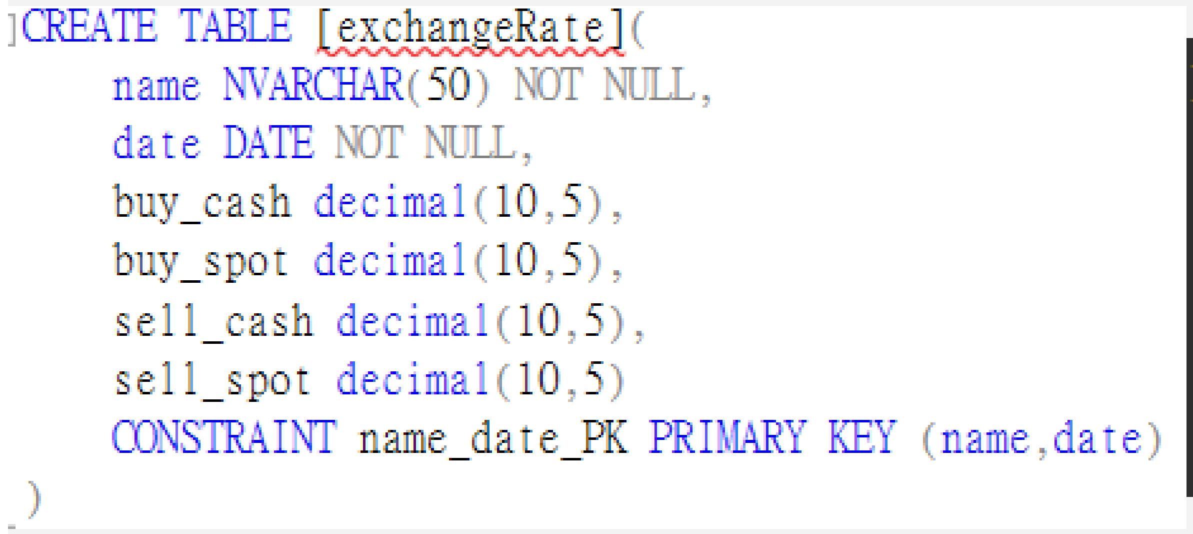
牌價最新掛牌時間：2024/04/24 14:35

幣別	現金匯率		即期匯率		遠期匯率	歷史匯率
	本行買入	本行賣出	本行買入	本行賣出		
美金 (USD)	32.11	32.78	32.46	32.56	查詢	查詢
港幣 (HKD)	3.994	4.198	4.12	4.18	查詢	查詢
英鎊 (GBP)	39.27	41.39	40.28	40.68	查詢	查詢
澳幣 (AUD)	20.78	21.56	21.07	21.27	查詢	查詢
加拿大幣 (CAD)	23.29	24.2	23.69	23.89	查詢	查詢
新加坡幣 (SGD)	23.32	24.23	23.81	23.99	查詢	查詢
瑞士法郎 (CHF)	34.81	36.01	35.49	35.74	查詢	查詢
日圓 (JPY)	0.2006	0.2134	0.2079	0.2119	查詢	查詢
南非幣 (ZAR)	-	-	1.662	1.742	查詢	查詢
瑞典幣 (SEK)	2.62	3.14	2.96	3.06	查詢	查詢
紐元 (NZD)	18.83	19.68	19.21	19.41	查詢	查詢
泰幣 (THB)	0.7532	0.9432	0.8656	0.9056	查詢	查詢
菲國比索 (PHP)	0.5004	0.6324	-	-	查詢	查詢
印尼幣 (IDR)	0.00168	0.00238	-	-	查詢	查詢
歐元 (EUR)	33.97	35.31	34.59	34.99	查詢	查詢
韓元 (KRW)	0.022	0.0259	-	-	查詢	查詢
越南盾 (VND)	0.00105	0.00146	-	-	查詢	查詢
馬來幣 (MYR)	5.766	7.286	-	-	查詢	查詢
人民幣 (CNY)	4.379	4.541	4.451	4.501	查詢	查詢

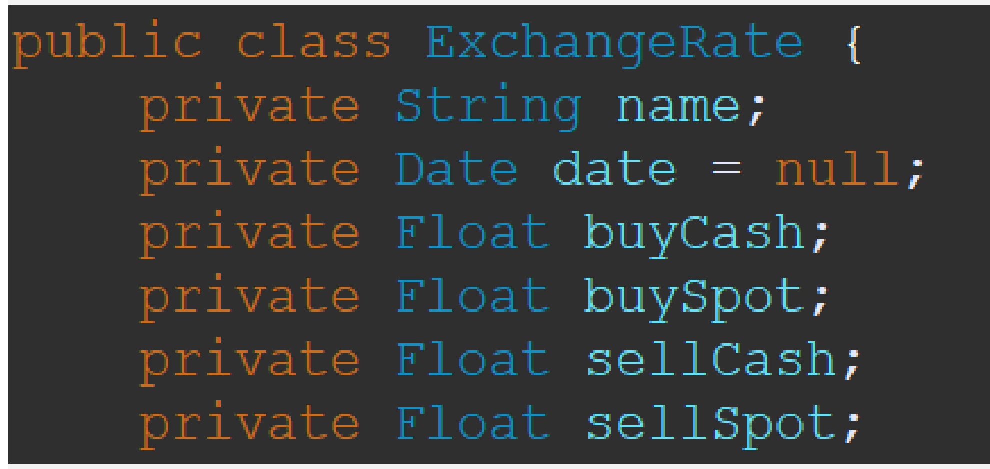
資料庫與名詞介紹

專題報告 - 匯率資料庫

欄位介紹



```
CREATE TABLE [exchangeRate](
    name NVARCHAR(50) NOT NULL,
    date DATE NOT NULL,
    buy_cash decimal(10,5),
    buy_spot decimal(10,5),
    sell_cash decimal(10,5),
    sell_spot decimal(10,5)
    CONSTRAINT name_date_PK PRIMARY KEY (name,date)
)
```



```
public class ExchangeRate {
    private String name;
    private Date date = null;
    private Float buyCash;
    private Float buySpot;
    private Float sellCash;
    private Float sellSpot;
```

名詞解釋

- Buy Cash：現金匯率 本行買入
- Buy Spot：即期匯率 本行買入
- Sell Cash：現金匯率 本行賣出
- Sell Spot：即期匯率 本行賣出

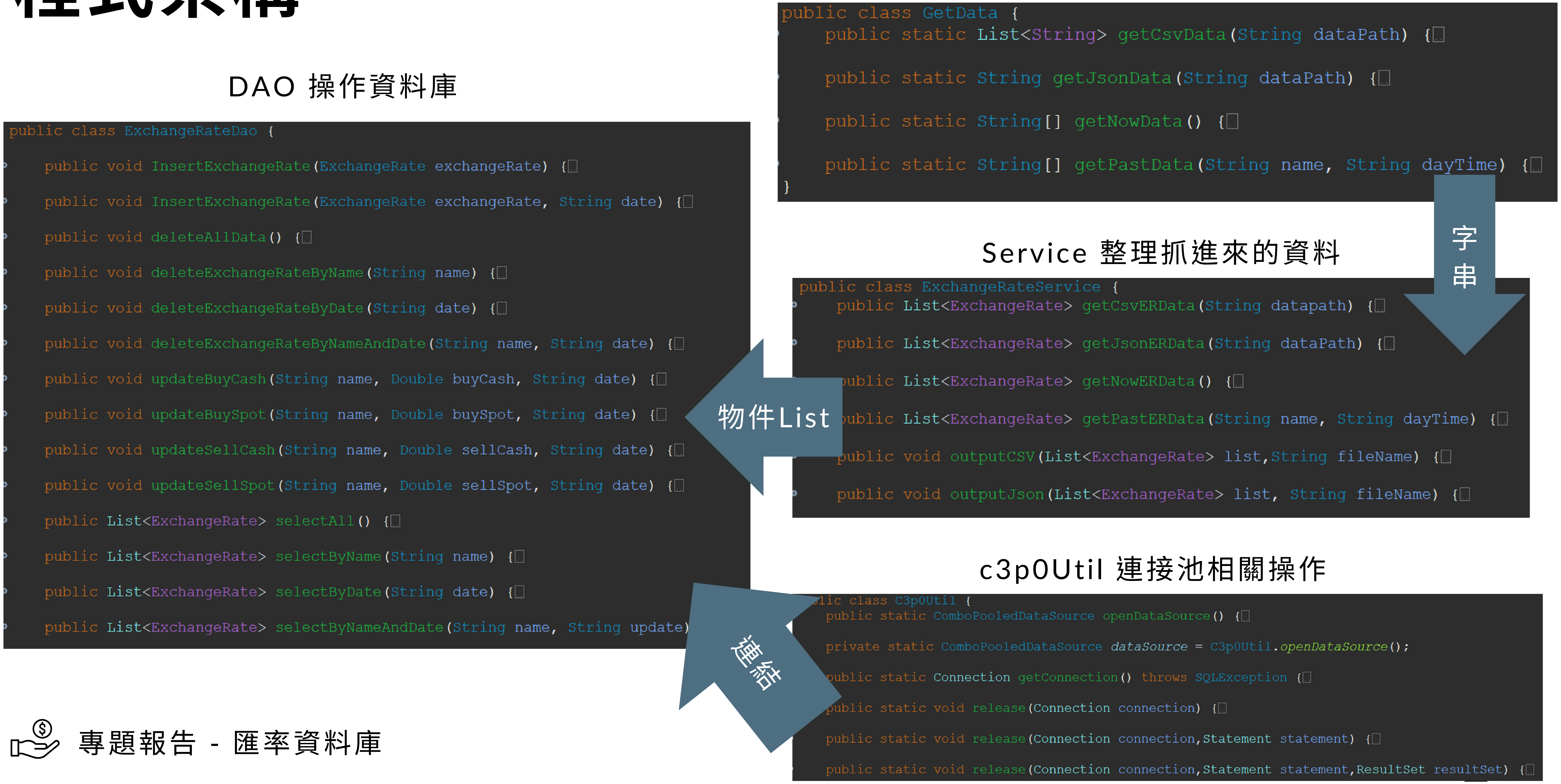
- 現金匯率：使用現金交易外匯
- 即期匯率：直接在帳戶中進行交易
- 本行買入：出售外幣給銀行的價格
- 本行賣出：向銀行買入外幣的價格

資料來源

臺灣銀行牌告匯率

此資料僅供報告用途，無商業行為

程式架構



GetData 抓資料

```
public class GetData {
    public static List<String> getCsvData(String dataPath) {□

    public static String getJsonData(String dataPath) {□

    public static String[] getNowData() {□

    public static String[] getPastData(String name, String dayTime) {□
}
```

DAO 操作資料庫

```
public class ExchangeRateDao {
    public void InsertExchangeRate(ExchangeRate exchangeRate) {□

    public void InsertExchangeRate(ExchangeRate exchangeRate, String date) {□

    public void deleteAllData() {□

    public void deleteExchangeRateByName(String name) {□

    public void deleteExchangeRateByDate(String date) {□

    public void deleteExchangeRateByNameAndDate(String name, String date) {□

    public void updateBuyCash(String name, Double buyCash, String date) {□

    public void updateBuySpot(String name, Double buySpot, String date) {□

    public void updateSellCash(String name, Double sellCash, String date) {□

    public void updateSellSpot(String name, Double sellSpot, String date) {□

    public List<ExchangeRate> selectAll() {□

    public List<ExchangeRate> selectByName(String name) {□

    public List<ExchangeRate> selectByDate(String date) {□

    public List<ExchangeRate> selectByNameAndDate(String name, String update)
```

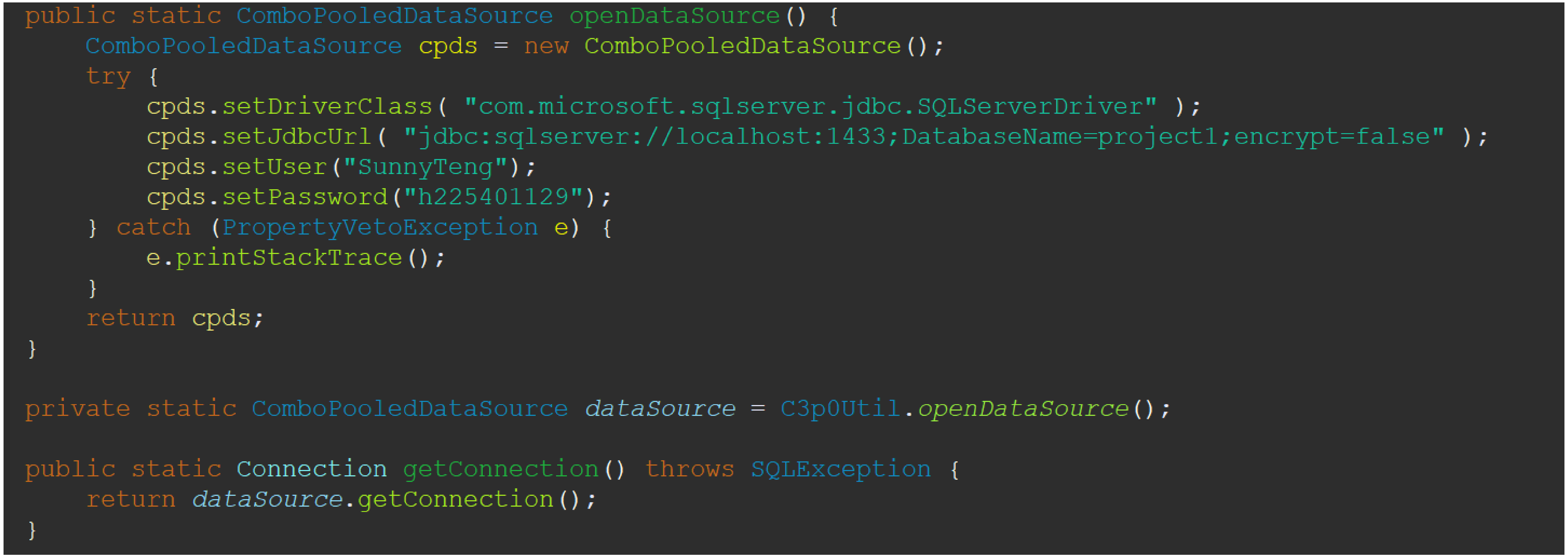
Service 整理抓進來的資料

```
public class ExchangeRateService {
    public List<ExchangeRate> getCsvERData(String datapath) {□

    public List<ExchangeRate> getJsonERData(String dataPath) {□

    ublic List<ExchangeRate> getNowERData() {□

    ublic List<ExchangeRate> getPastERData(String name, String dayTime) {□

    public void outputCSV(List<ExchangeRate> list,String fileName) {□

    public void outputJson(List<ExchangeRate> list, String fileName) {□
```

c3p0Util 連接池相關操作

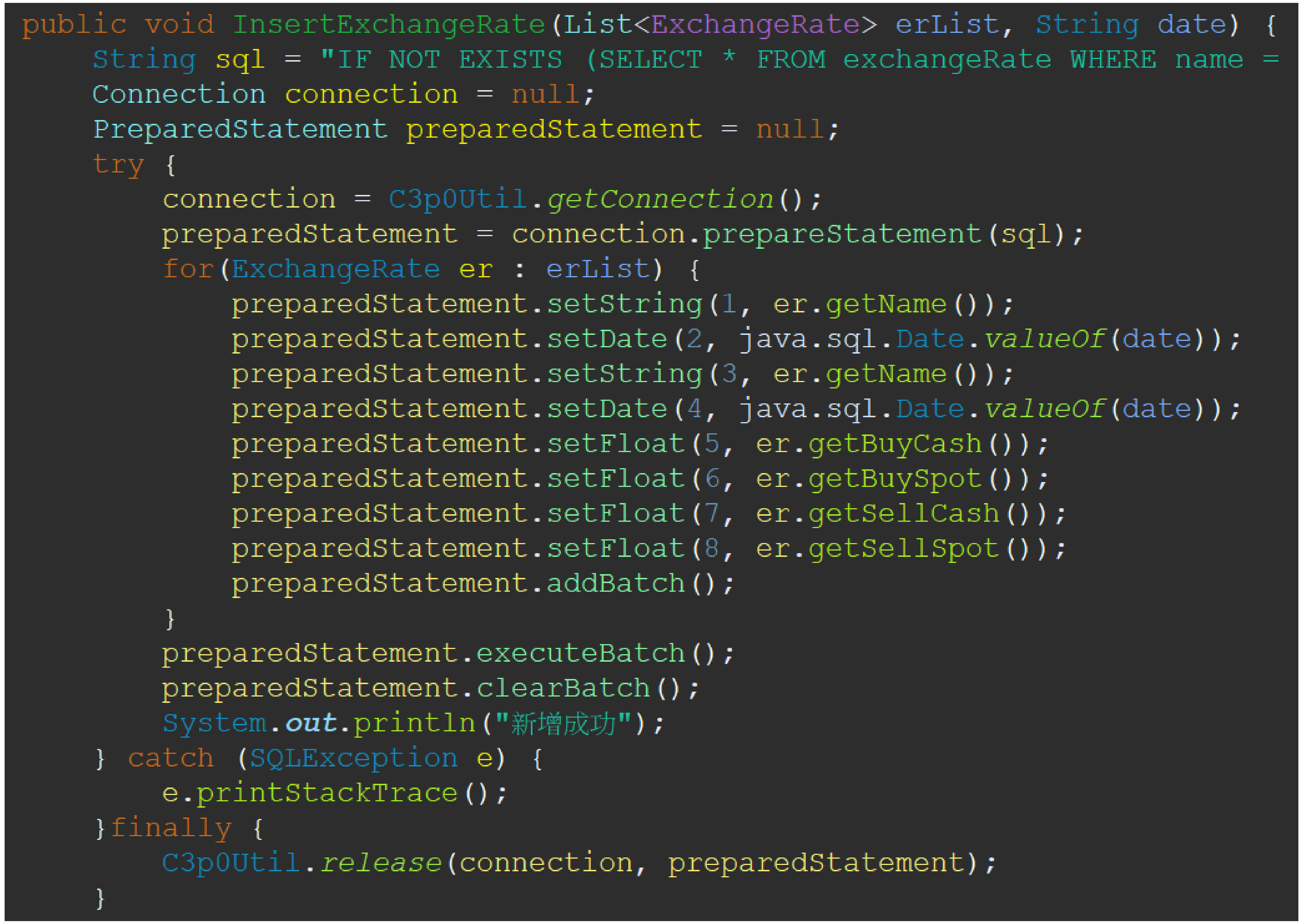
```
lic class C3p0Util {
    public static ComboPooledDataSource openDataSource() {□

    private static ComboPooledDataSource dataSource = C3p0Util.openDataSource();

    public static Connection getConnection() throws SQLException {□

    public static void release(Connection connection) {□

    public static void release(Connection connection,Statement statement) {□

    public static void release(Connection connection,Statement statement,ResultSet resultSet) {□
```

CONNECTION POOL



```
public static ComboPooledDataSource openDataSource() {
    ComboPooledDataSource cpds = new ComboPooledDataSource();
    try {
        cpds.setDriverClass( "com.microsoft.sqlserver.jdbc.SQLServerDriver" );
        cpds.setJdbcUrl( "jdbc:sqlserver://localhost:1433;DatabaseName=project1;encrypt=false" );
        cpds.setUser("SunnyTeng");
        cpds.setPassword("h225401129");
    } catch (PropertyVetoException e) {
        e.printStackTrace();
    }
    return cpds;
}

private static ComboPooledDataSource dataSource = C3p0Util.openDataSource();

public static Connection getConnection() throws SQLException {
    return dataSource.getConnection();
}
```

INSERT - BATCH

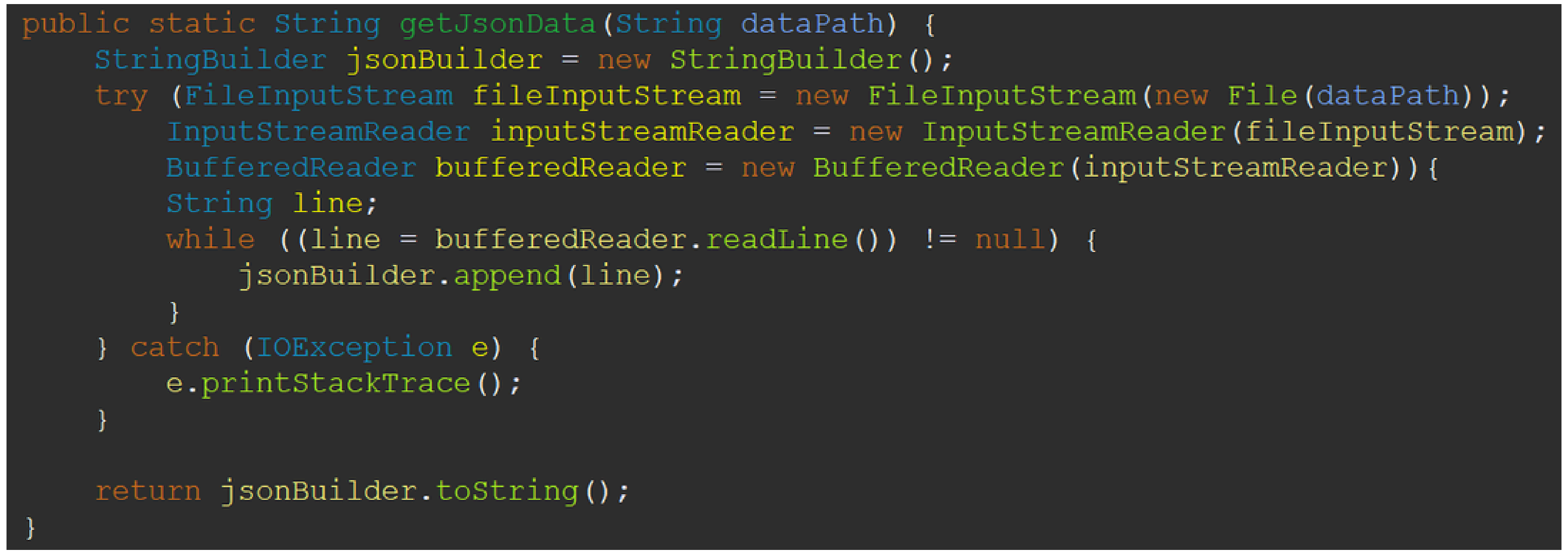


```java
public void InsertExchangeRate(List<ExchangeRate> erList, String date) {
    String sql = "IF NOT EXISTS (SELECT * FROM exchangeRate WHERE name =
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    try {
        connection = C3p0Util.getConnection();
        preparedStatement = connection.prepareStatement(sql);
        for(ExchangeRate er : erList) {
            preparedStatement.setString(1, er.getName());
            preparedStatement.setDate(2, java.sql.Date.valueOf(date));
            preparedStatement.setString(3, er.getName());
            preparedStatement.setDate(4, java.sql.Date.valueOf(date));
            preparedStatement.setFloat(5, er.getBuyCash());
            preparedStatement.setFloat(6, er.getBuySpot());
            preparedStatement.setFloat(7, er.getSellCash());
            preparedStatement.setFloat(8, er.getSellSpot());
            preparedStatement.addBatch();
        }
        preparedStatement.executeBatch();
        preparedStatement.clearBatch();
        System.out.println("新增成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }finally {
        C3p0Util.release(connection, preparedStatement);
    }
```

爬蟲

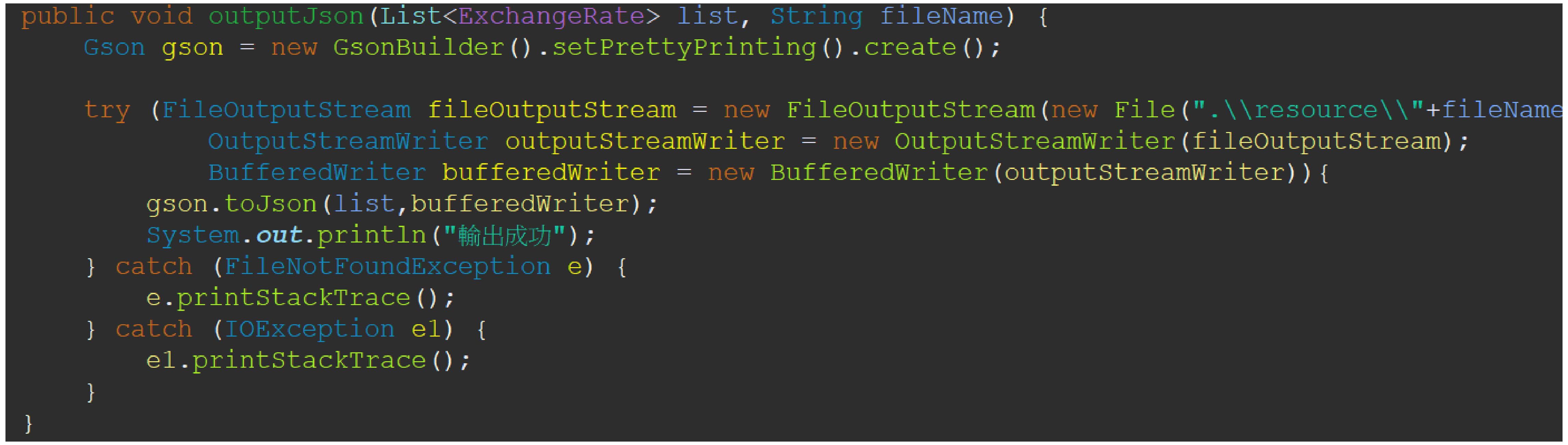
```java
public static String[] getNowData() {
    String[] dataList = null;
    try {
        Document document = Jsoup.connect("https://rate.bot.com.tw/xrt/flcsv/0/day").get();
        String body = document.body().text();
        dataList = body.split(" ");
    } catch (IOException e) {
        e.printStackTrace();
    }
    return dataList;
}

public static String[] getPastData(String name, String dayTime) {
    String[] dataList = null;
    try {
        Document document = Jsoup.connect("https://rate.bot.com.tw/xrt/flcsv/0/"+dayTime+"/"+name).get();
        String body = document.body().text();
        dataList = body.split(" ");
    } catch (IOException e) {
        e.printStackTrace();
    }
    return dataList;
}
```

IO - JSON輸入



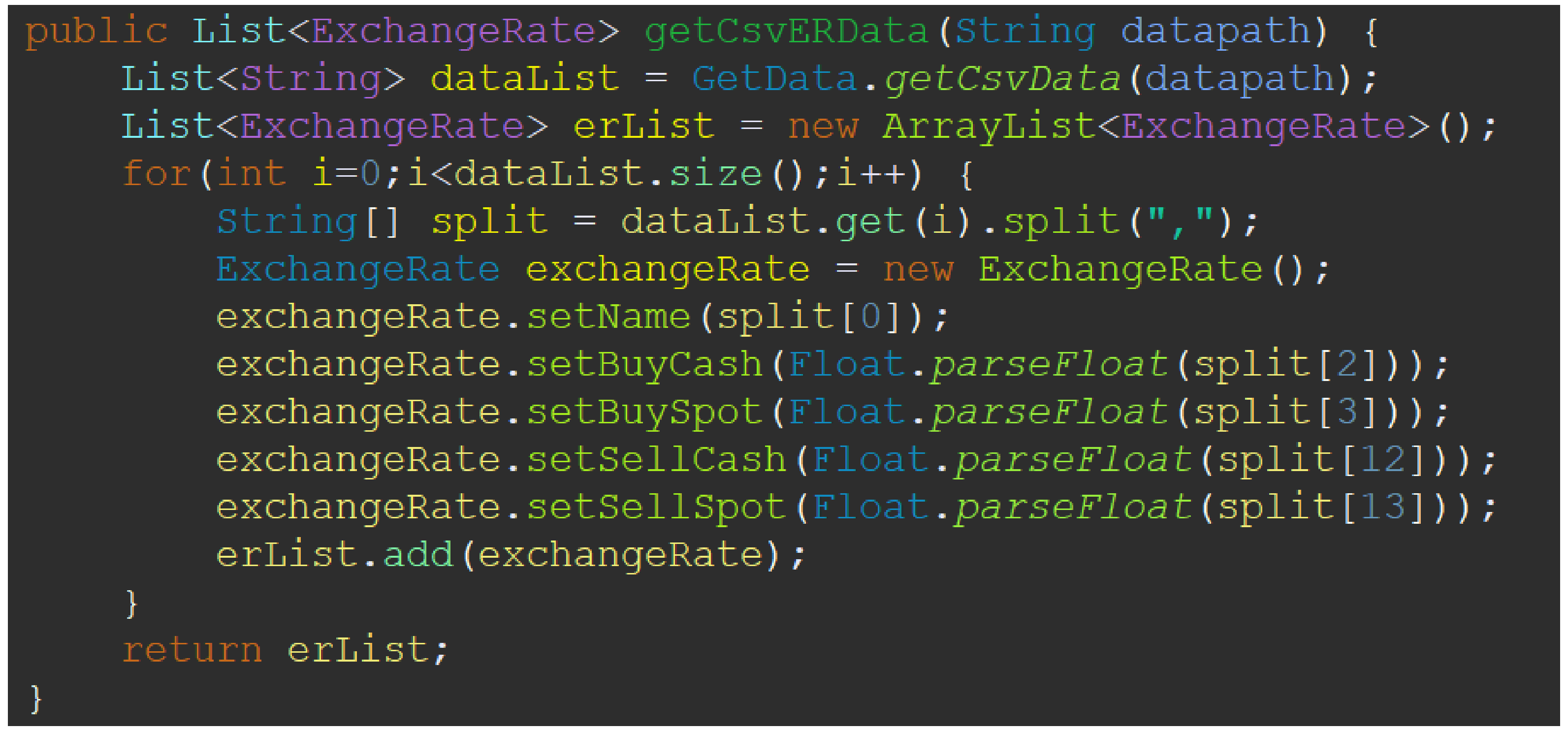
```java
public static String getJsonData(String dataPath) {
    StringBuilder jsonBuilder = new StringBuilder();
    try (FileInputStream fileInputStream = new FileInputStream(new File(dataPath));
        InputStreamReader inputStreamReader = new InputStreamReader(fileInputStream);
        BufferedReader bufferedReader = new BufferedReader(inputStreamReader)){
        String line;
        while ((line = bufferedReader.readLine()) != null) {
            jsonBuilder.append(line);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }

    return jsonBuilder.toString();
}
```

IO - JSON輸出



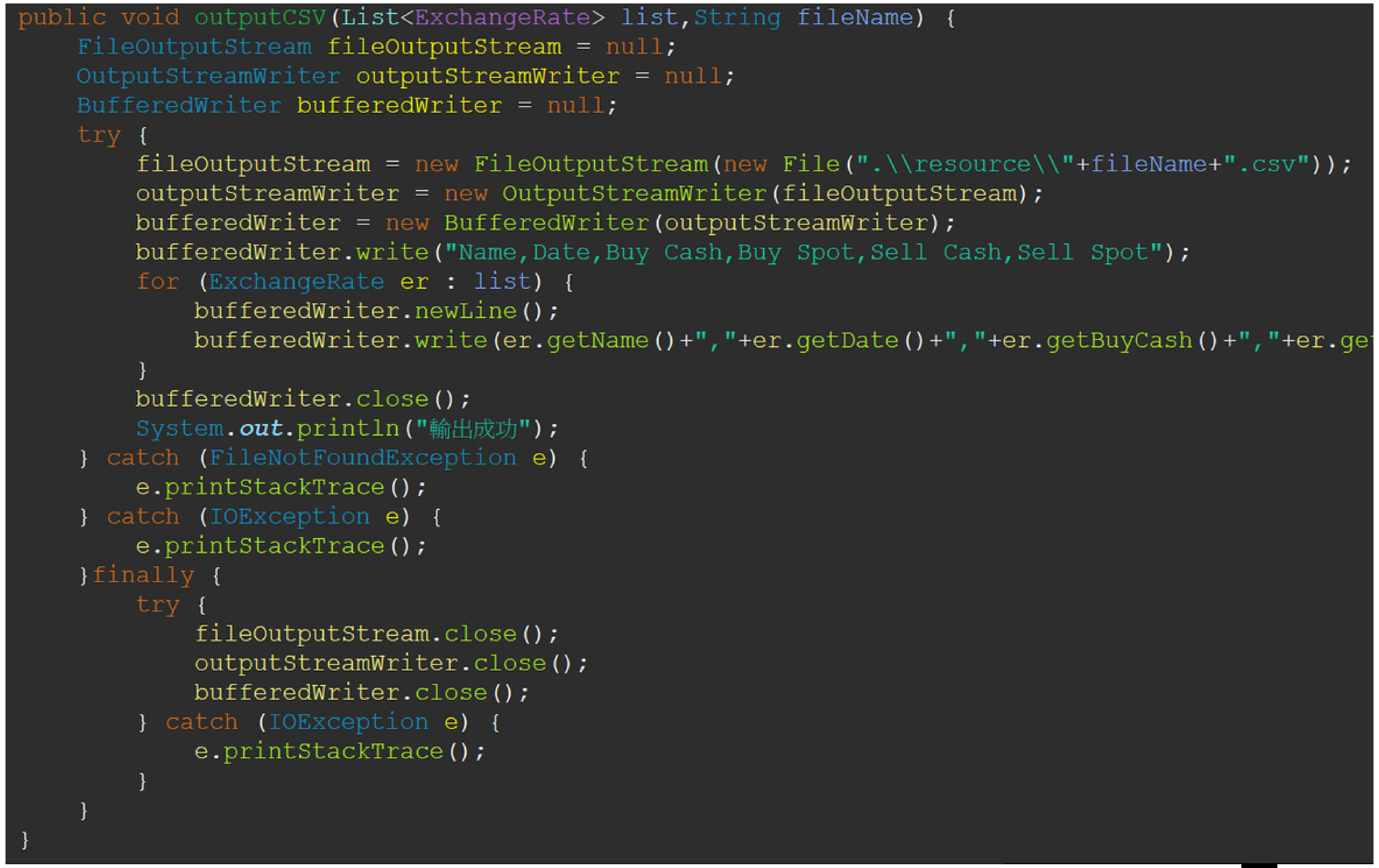
```java
public void outputJson(List<ExchangeRate> list, String fileName) {
    Gson gson = new GsonBuilder().setPrettyPrinting().create();

    try (FileOutputStream fileOutputStream = new FileOutputStream(new File(".\\resource\\"+fileName
            OutputStreamWriter outputStreamWriter = new OutputStreamWriter(fileOutputStream);
            BufferedWriter bufferedWriter = new BufferedWriter(outputStreamWriter)){
        gson.toJson(list,bufferedWriter);
        System.out.println("輸出成功");
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e1) {
        e1.printStackTrace();
    }
}
```

IO - CSV輸入



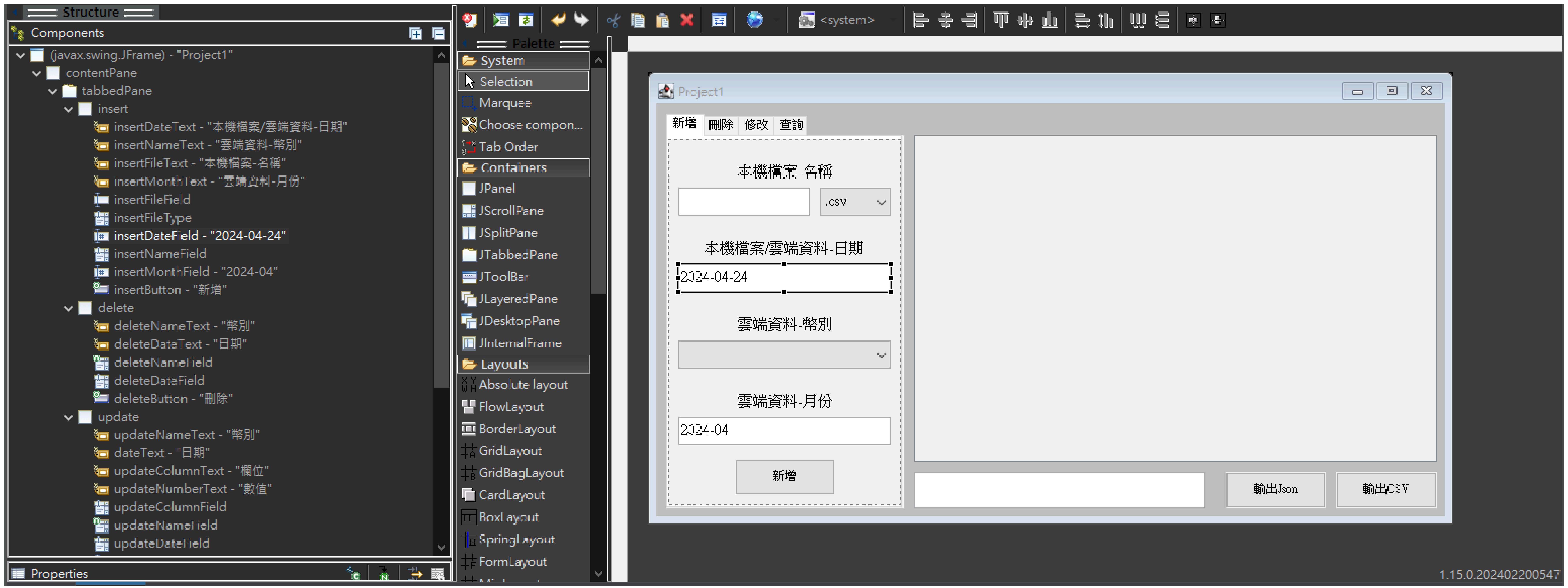
```java
public List<ExchangeRate> getCsvERData(String datapath) {
    List<String> dataList = GetData.getCsvData(datapath);
    List<ExchangeRate> erList = new ArrayList<ExchangeRate>();
    for(int i=0;i<dataList.size();i++) {
        String[] split = dataList.get(i).split(",");
        ExchangeRate exchangeRate = new ExchangeRate();
        exchangeRate.setName(split[0]);
        exchangeRate.setBuyCash(Float.parseFloat(split[2]));
        exchangeRate.setBuySpot(Float.parseFloat(split[3]));
        exchangeRate.setSellCash(Float.parseFloat(split[12]));
        exchangeRate.setSellSpot(Float.parseFloat(split[13]));
        erList.add(exchangeRate);
    }
    return erList;
}
```

IO - CSV輸出



```java
public void outputCSV(List<ExchangeRate> list,String fileName) {
    FileOutputStream fileOutputStream = null;
    OutputStreamWriter outputStreamWriter = null;
    BufferedWriter bufferedWriter = null;
    try {
        fileOutputStream = new FileOutputStream(new File(".\\resource\\"+fileName+".csv"));
        outputStreamWriter = new OutputStreamWriter(fileOutputStream);
        bufferedWriter = new BufferedWriter(outputStreamWriter);
        bufferedWriter.write("Name,Date,Buy Cash,Buy Spot,Sell Cash,Sell Spot");
        for (ExchangeRate er : list) {
            bufferedWriter.newLine();
            bufferedWriter.write(er.getName()+","+er.getDate()+","+er.getBuyCash()+","+er.ge
        }
        bufferedWriter.close();
        System.out.println("輸出成功");
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }finally {
        try {
            fileOutputStream.close();
            outputStreamWriter.close();
            bufferedWriter.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

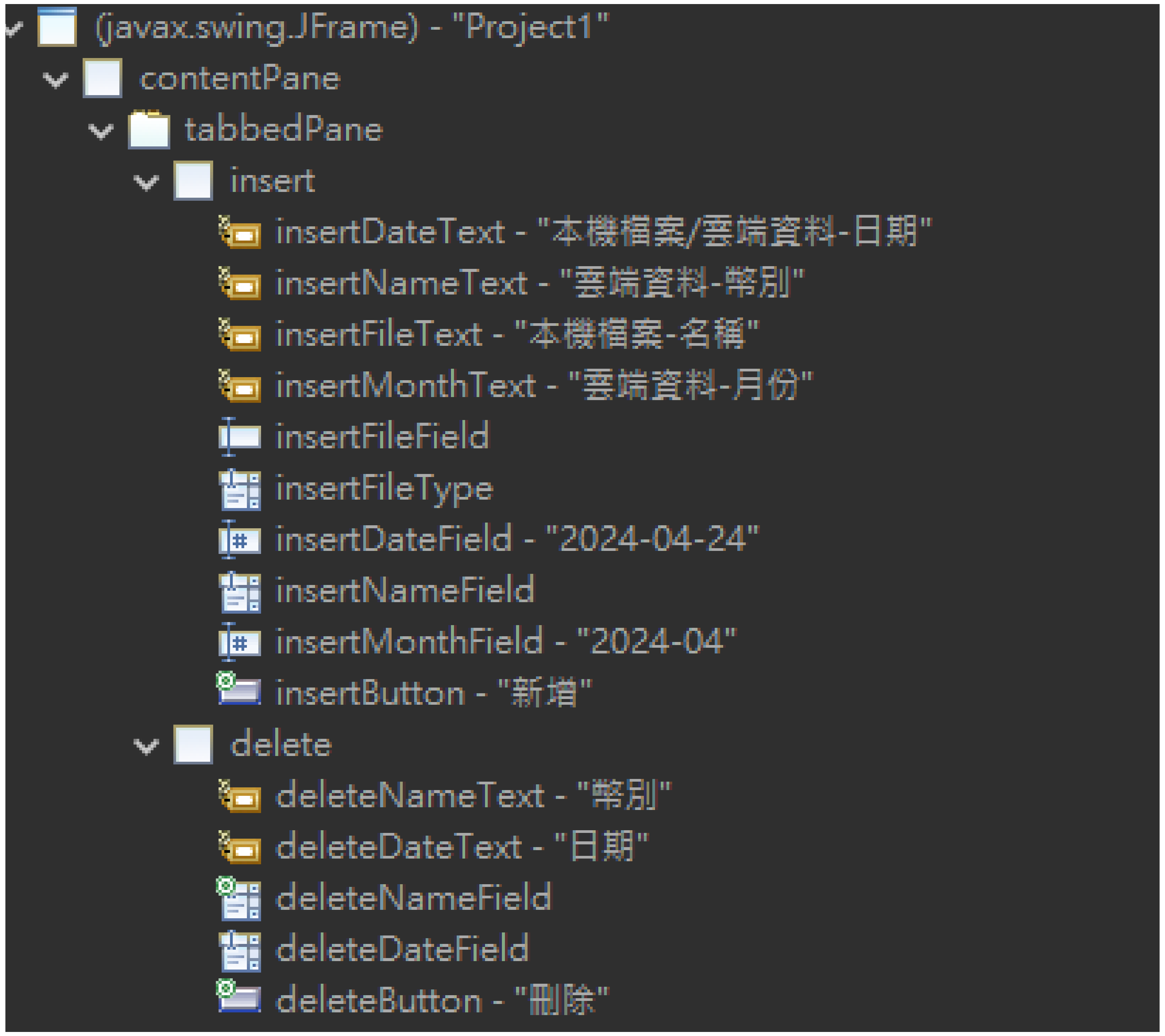
SWING - WINDOWBUILDER



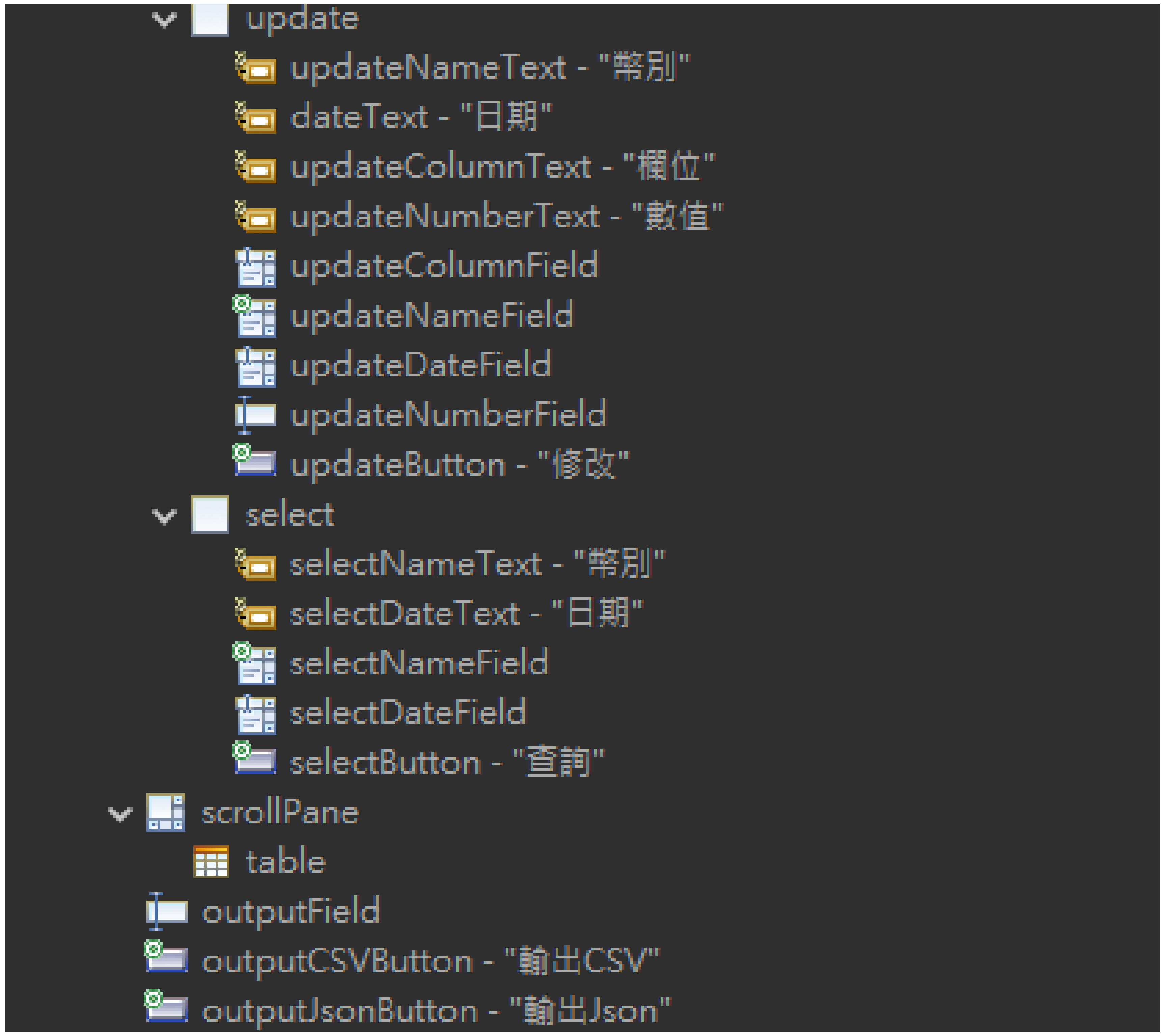
```
Structure
Components
(javax.swing.JFrame) - "Project1"
  contentPane
    tabbedPane
      insert
        insertDateText - "本機檔案/雲端資料-日期"
        insertNameText - "雲端資料-幣別"
        insertFileText - "本機檔案-名稱"
        insertMonthText - "雲端資料-月份"
        insertFileField
        insertFileType
        insertDateField - "2024-04-24"
        insertNameField
        insertMonthField - "2024-04"
        insertButton - "新增"
      delete
        deleteNameText - "幣別"
        deleteDateText - "日期"
        deleteNameField
        deleteDateField
        deleteButton - "刪除"
      update
        updateNameText - "幣別"
        dateText - "日期"
        updateColumnText - "欄位"
        updateNumberText - "數值"
        updateColumnField
        updateNameField
        updateDateField
Properties
```

專題報告 - 匯率資料庫

SWING - WINDOWBUILDER



```
(javax.swing.JFrame) - "Project1"
  contentPane
    tabbedPane
      insert
        insertDateText - "本機檔案/雲端資料-日期"
        insertNameText - "雲端資料-幣別"
        insertFileText - "本機檔案-名稱"
        insertMonthText - "雲端資料-月份"
        insertFileField
        insertFileType
        insertDateField - "2024-04-24"
        insertNameField
        insertMonthField - "2024-04"
        insertButton - "新增"
      delete
        deleteNameText - "幣別"
        deleteDateText - "日期"
        deleteNameField
        deleteDateField
        deleteButton - "刪除"
```

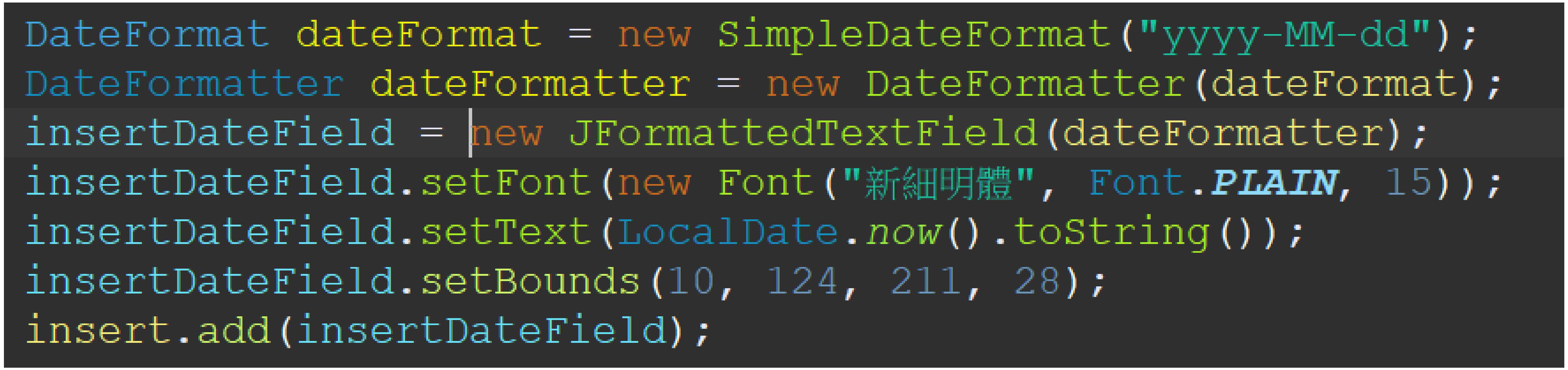


```
      update
        updateNameText - "幣別"
        dateText - "日期"
        updateColumnText - "欄位"
        updateNumberText - "數值"
        updateColumnField
        updateNameField
        updateDateField
        updateNumberField
        updateButton - "修改"
      select
        selectNameText - "幣別"
        selectDateText - "日期"
        selectNameField
        selectDateField
        selectButton - "查詢"
    scrollPane
      table
    outputField
    outputCSVButton - "輸出CSV"
    outputJsonButton - "輸出Json"
```

優化使用者體驗

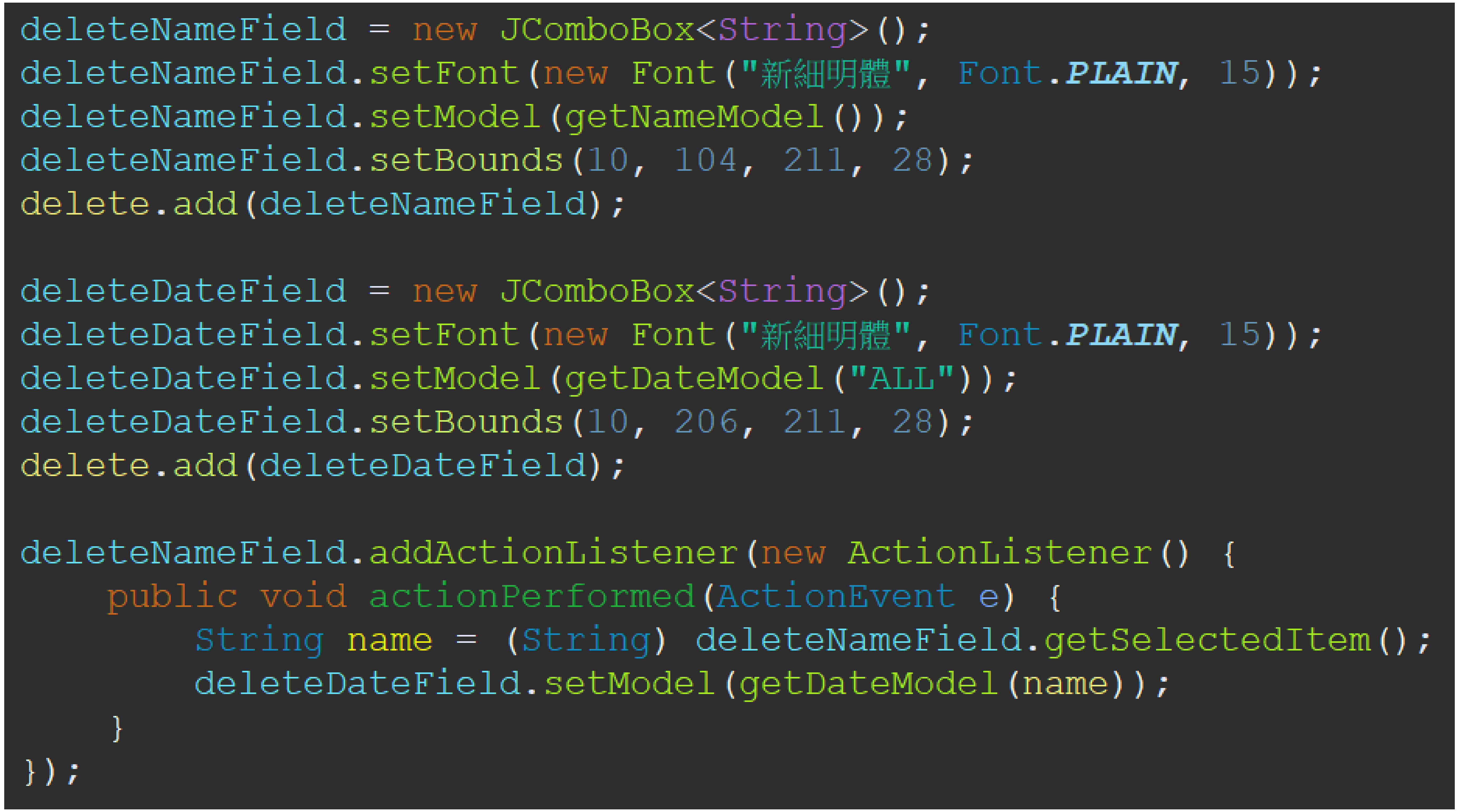
- 日期/月份欄位指定格式：防止使用者輸入錯誤
- 盡量使用下拉選單(JcomboBox)：減少打字的錯誤和麻煩
- 偵測選擇的幣別自動更新日期：透過下拉選單即可得知資料庫現存資料
- 操作提示對話框：讓使用者知道操作是否順利完成
- 增刪改後自動SelectAll：減少操作步驟

日期欄位指定格式



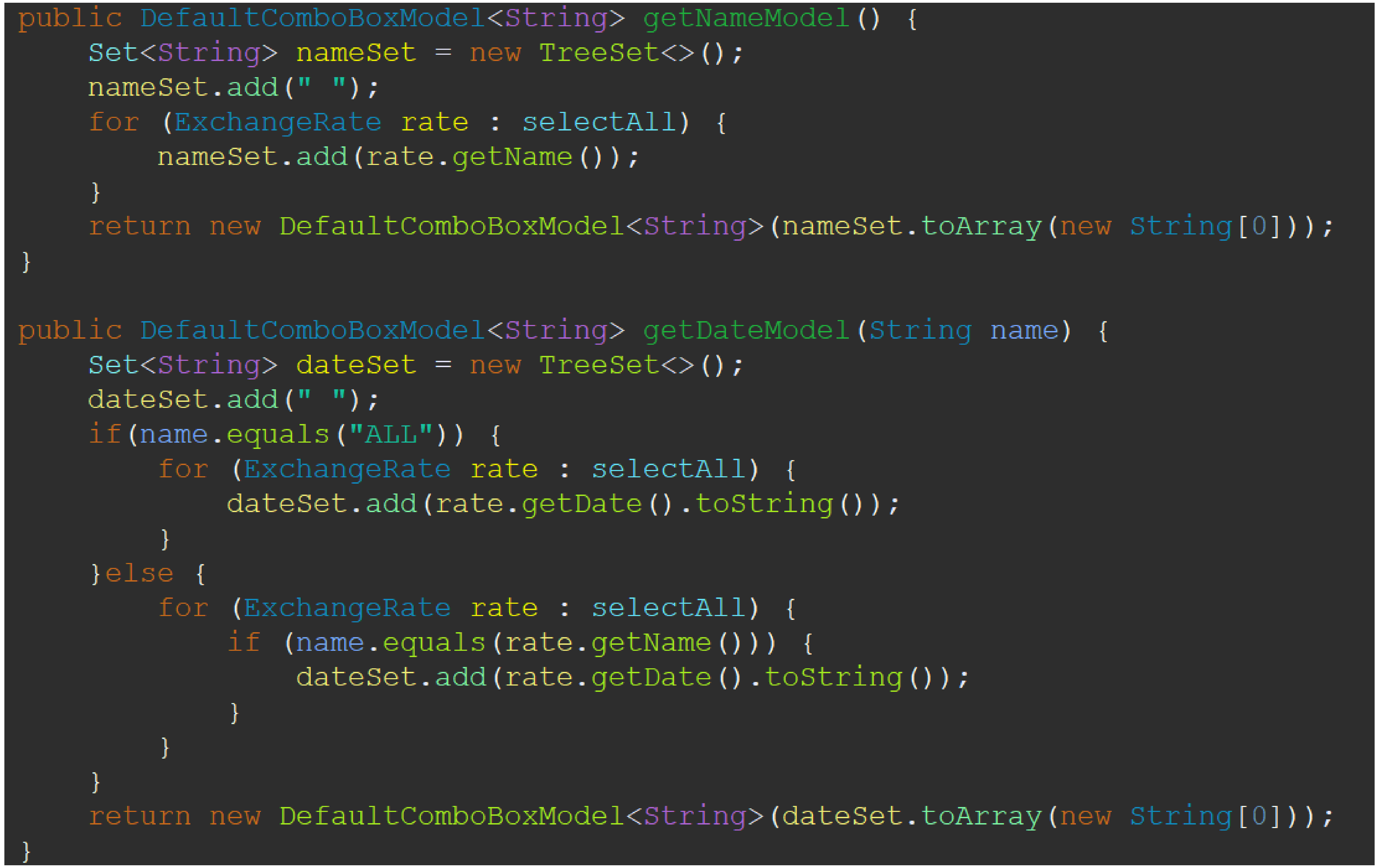
```
DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
DateFormatter dateFormatter = new DateFormatter(dateFormat);
insertDateField = new JFormattedTextField(dateFormatter);
insertDateField.setFont(new Font("新細明體", Font.PLAIN, 15));
insertDateField.setText(LocalDate.now().toString());
insertDateField.setBounds(10, 124, 211, 28);
insert.add(insertDateField);
```

偵測選擇的幣別　自動更新日期



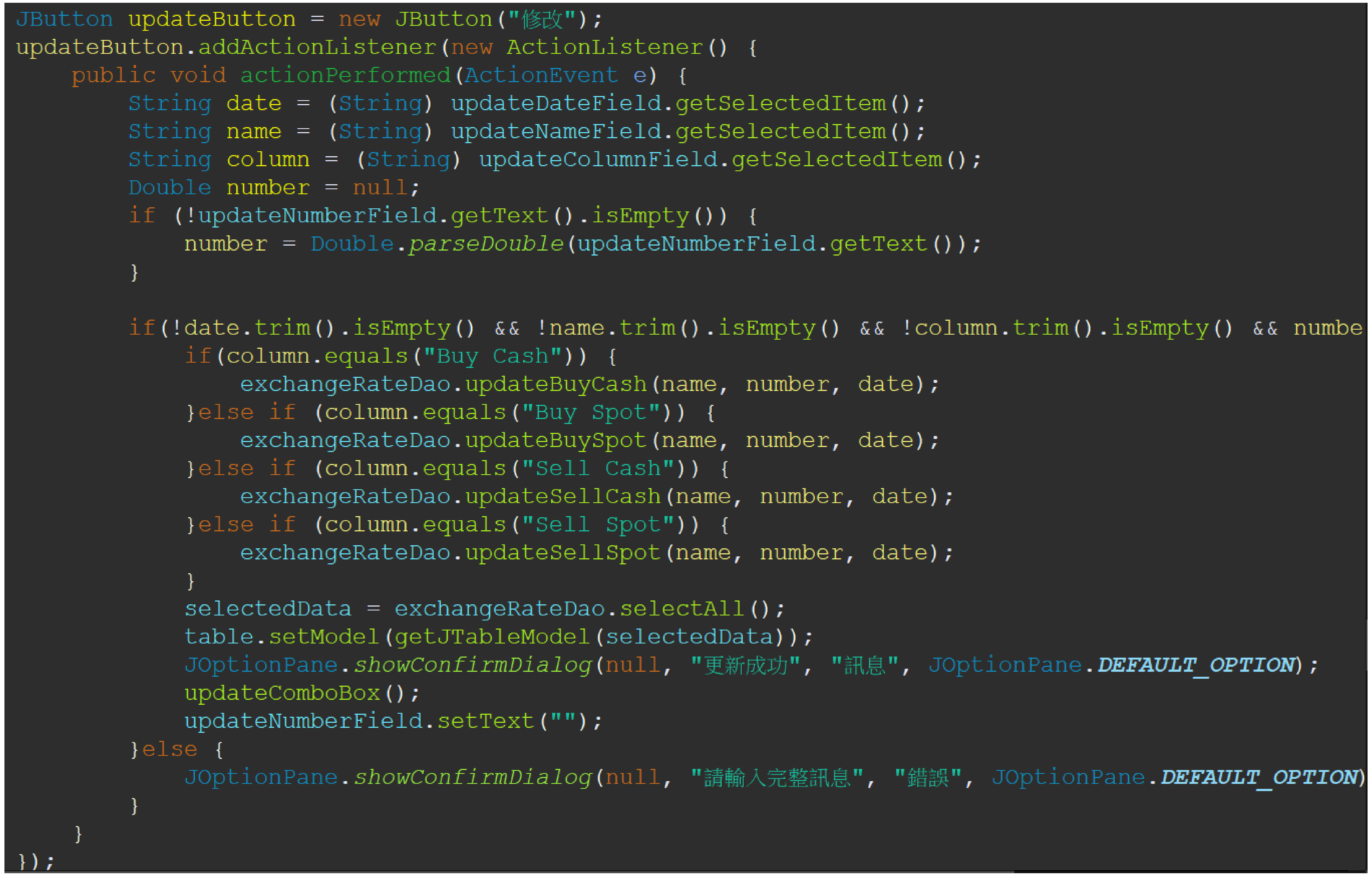
```java
deleteNameField = new JComboBox<String>();
deleteNameField.setFont(new Font("新細明體", Font.PLAIN, 15));
deleteNameField.setModel(getNameModel());
deleteNameField.setBounds(10, 104, 211, 28);
delete.add(deleteNameField);

deleteDateField = new JComboBox<String>();
deleteDateField.setFont(new Font("新細明體", Font.PLAIN, 15));
deleteDateField.setModel(getDateModel("ALL"));
deleteDateField.setBounds(10, 206, 211, 28);
delete.add(deleteDateField);

deleteNameField.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String name = (String) deleteNameField.getSelectedItem();
        deleteDateField.setModel(getDateModel(name));
    }
});
```

偵測選擇的幣別　自動更新日期



```java
public DefaultComboBoxModel<String> getNameModel() {
    Set<String> nameSet = new TreeSet<>();
    nameSet.add(" ");
    for (ExchangeRate rate : selectAll) {
        nameSet.add(rate.getName());
    }
    return new DefaultComboBoxModel<String>(nameSet.toArray(new String[0]));
}

public DefaultComboBoxModel<String> getDateModel(String name) {
    Set<String> dateSet = new TreeSet<>();
    dateSet.add(" ");
    if(name.equals("ALL")) {
        for (ExchangeRate rate : selectAll) {
            dateSet.add(rate.getDate().toString());
        }
    }else {
        for (ExchangeRate rate : selectAll) {
            if (name.equals(rate.getName())) {
                dateSet.add(rate.getDate().toString());
            }
        }
    }
    return new DefaultComboBoxModel<String>(dateSet.toArray(new String[0]));
}
```

按鈕邏輯 - 以UPDATE為例



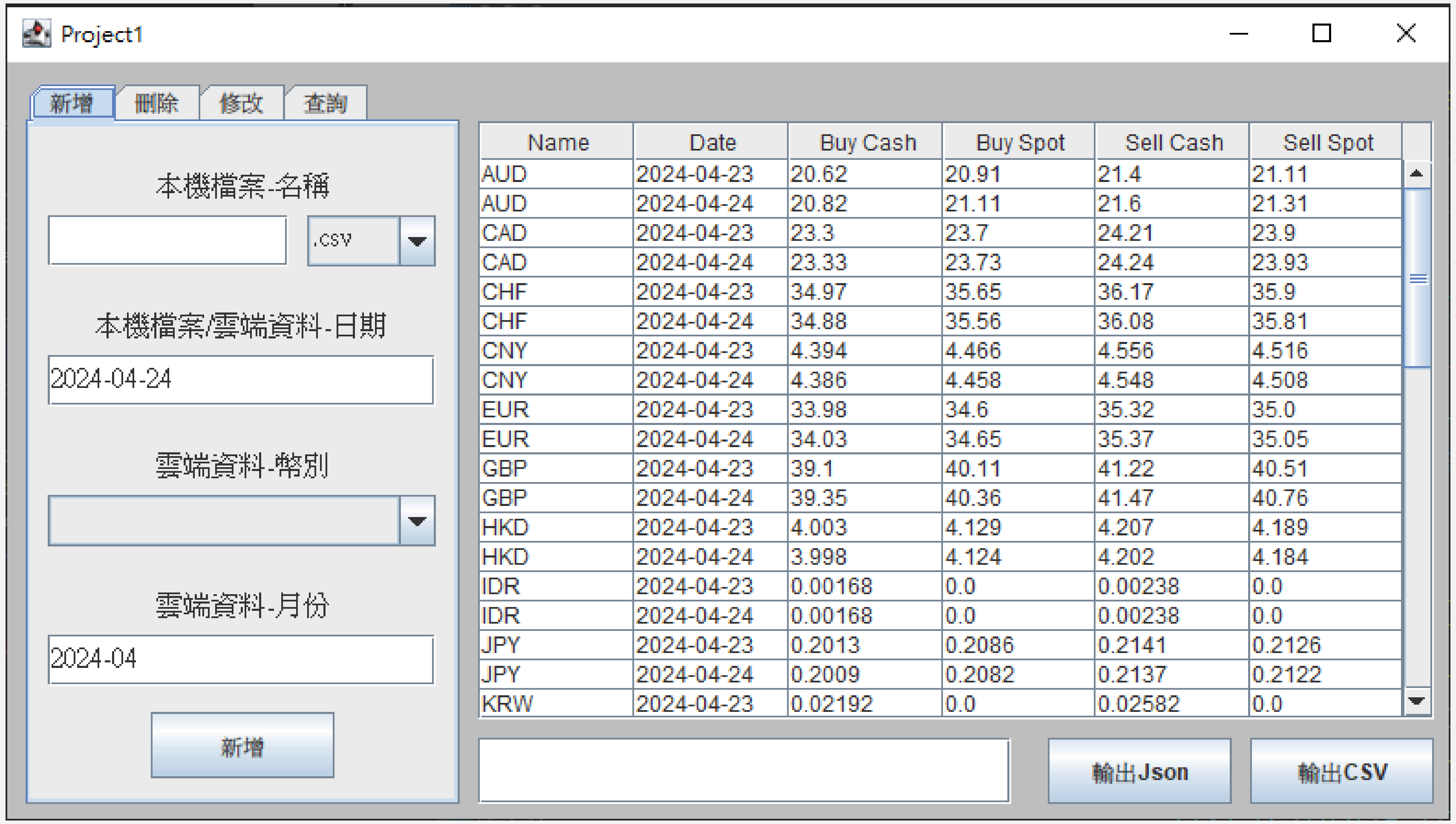
```java
JButton updateButton = new JButton("修改");
updateButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String date = (String) updateDateField.getSelectedItem();
        String name = (String) updateNameField.getSelectedItem();
        String column = (String) updateColumnField.getSelectedItem();
        Double number = null;
        if (!updateNumberField.getText().isEmpty()) {
            number = Double.parseDouble(updateNumberField.getText());
        }

        if(!date.trim().isEmpty() && !name.trim().isEmpty() && !column.trim().isEmpty() && numbe
            if(column.equals("Buy Cash")) {
                exchangeRateDao.updateBuyCash(name, number, date);
            }else if (column.equals("Buy Spot")) {
                exchangeRateDao.updateBuySpot(name, number, date);
            }else if (column.equals("Sell Cash")) {
                exchangeRateDao.updateSellCash(name, number, date);
            }else if (column.equals("Sell Spot")) {
                exchangeRateDao.updateSellSpot(name, number, date);
            }
            selectedData = exchangeRateDao.selectAll();
            table.setModel(getJTableModel(selectedData));
            JOptionPane.showConfirmDialog(null, "更新成功", "訊息", JOptionPane.DEFAULT_OPTION);
            updateComboBox();
            updateNumberField.setText("");
        }else {
            JOptionPane.showConfirmDialog(null, "請輸入完整訊息", "錯誤", JOptionPane.DEFAULT_OPTION)
        }
    }
});
```

功能總表

- 新增
    - 新增本機資料
        - CSV檔
        - JSON檔
    - 新增網路資料
        - 依日期新增
        - 依幣別新增月資料
- 刪除
    - 刪除所有
    - 依幣別刪除
    - 依日期刪除
- 修改
    - 依指定欄位修改
- 查詢
    - 查詢所有
    - 依幣別查詢
    - 依日期查詢
- 輸出
    - 輸出CSV
    - 輸出JSON

成果展示

```
java -Dfile.encoding=utf-8 -jar demo.jar
```



本機檔案-名稱

.csv

本機檔案/雲端資料-日期

2024-04-24

雲端資料-幣別

雲端資料-月份

2024-04

Name	Date	Buy Cash	Buy Spot	Sell Cash	Sell Spot
AUD	2024-04-23	20.62	20.91	21.4	21.11
AUD	2024-04-24	20.82	21.11	21.6	21.31
CAD	2024-04-23	23.3	23.7	24.21	23.9
CAD	2024-04-24	23.33	23.73	24.24	23.93
CHF	2024-04-23	34.97	35.65	36.17	35.9
CHF	2024-04-24	34.88	35.56	36.08	35.81
CNY	2024-04-23	4.394	4.466	4.556	4.516
CNY	2024-04-24	4.386	4.458	4.548	4.508
EUR	2024-04-23	33.98	34.6	35.32	35.0
EUR	2024-04-24	34.03	34.65	35.37	35.05
GBP	2024-04-23	39.1	40.11	41.22	40.51
GBP	2024-04-24	39.35	40.36	41.47	40.76
HKD	2024-04-23	4.003	4.129	4.207	4.189
HKD	2024-04-24	3.998	4.124	4.202	4.184
IDR	2024-04-23	0.00168	0.0	0.00238	0.0
IDR	2024-04-24	0.00168	0.0	0.00238	0.0
JPY	2024-04-23	0.2013	0.2086	0.2141	0.2126
JPY	2024-04-24	0.2009	0.2082	0.2137	0.2122
KRW	2024-04-23	0.02192	0.0	0.02582	0.0