

## Chuyên đề 3

### Xử lý đa luồng

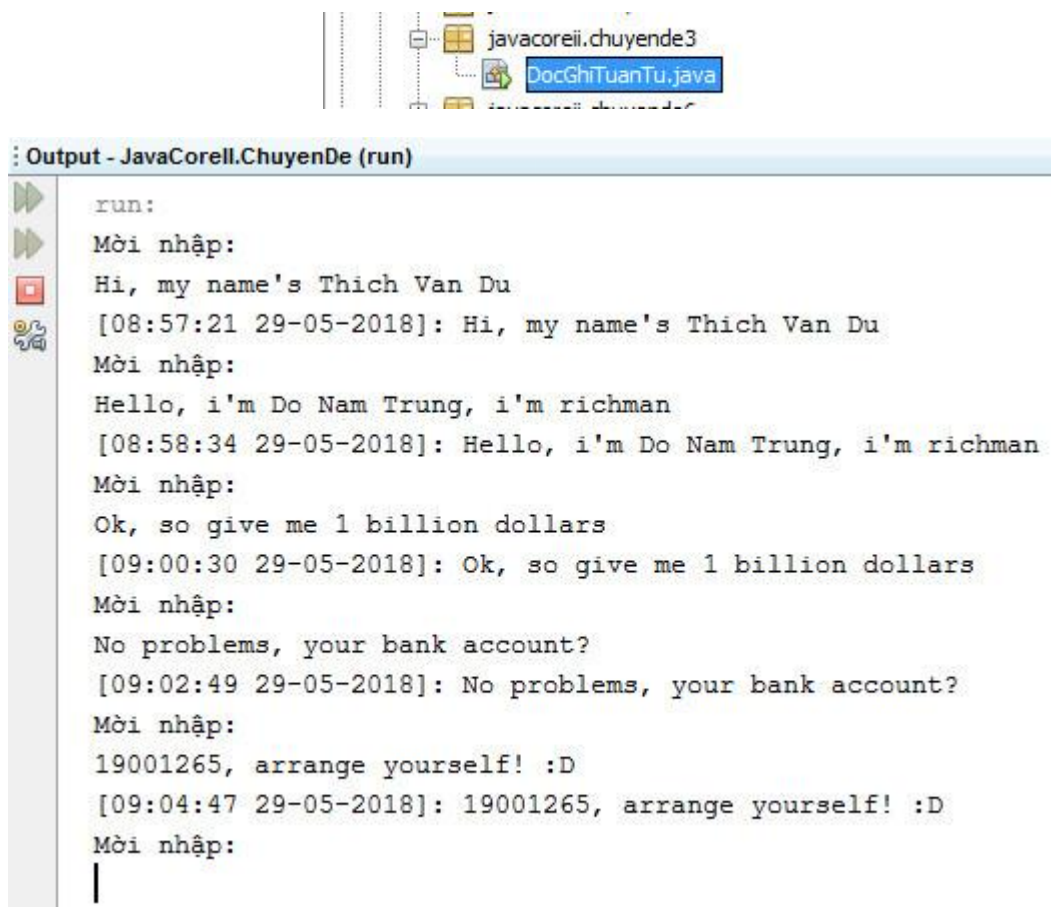
#### Mục tiêu

- ✓ Tạo được luồng bằng 2 cách: Thread và Runnable
- ✓ Vận dụng được kiến thức về đồng bộ (synchronize) luồng để xử lý dữ liệu dùng chung.

#### Bài thực hành 1:

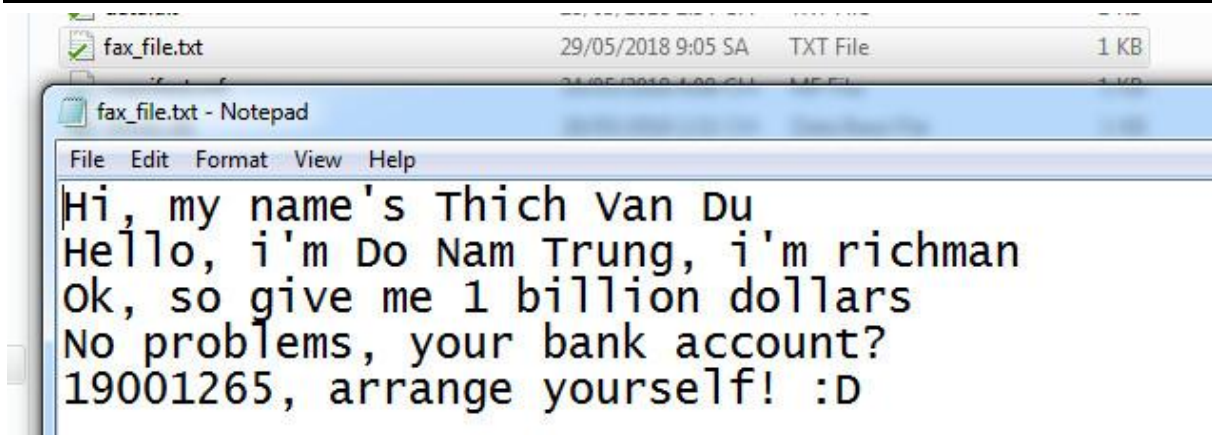
**Yêu cầu:** Viết chương trình Java sử dụng Thread để thực thi song song công việc:

- Luồng 1: nhập dữ liệu (chuỗi văn bản)
- Luồng 2: khi kết thúc nhập dữ liệu (nhấn enter) thì hiển thị nội dung trên lên màn hình kèm thời gian hiện tại đồng thời ghi dữ liệu đó ra file ***data.txt***.



```

run:
Mời nhập:
Hi, my name's Thich Van Du
[08:57:21 29-05-2018]: Hi, my name's Thich Van Du
Mời nhập:
Hello, i'm Do Nam Trung, i'm richman
[08:58:34 29-05-2018]: Hello, i'm Do Nam Trung, i'm richman
Mời nhập:
Ok, so give me 1 billion dollars
[09:00:30 29-05-2018]: Ok, so give me 1 billion dollars
Mời nhập:
No problems, your bank account?
[09:02:49 29-05-2018]: No problems, your bank account?
Mời nhập:
19001265, arrange yourself! :D
[09:04:47 29-05-2018]: 19001265, arrange yourself! :D
Mời nhập:
|
    
```

**Code tham khảo:****DocGhiTuanTu.java**

```
package javacoreii.chuyende3;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.DateFormat;
import java.util.Calendar;
import java.util.Locale;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author minhvufc
 */
public class DocGhiTuanTu {

    MayFax fax;

    private void demo() {
        fax = new MayFax();
        TaskDoc taskDoc = new TaskDoc(fax);
        TaskGhi taskGhi = new TaskGhi(fax);

        taskDoc.start();
        taskGhi.start();
    }

    /**
```

```

* Lớp inner class có 2 phương thức đọc và ghi dữ liệu
*/
private class MayFax {

    final String filePath = "fax_file.txt";
    String data;

    /**
     * Yêu cầu nhập dữ liệu
     */
    public void read() {
        System.out.println("Mời nhập: ");
        Scanner nhap = new Scanner(System.in);
        data = nhap.nextLine();
    }

    /**
     * Hiển thị dữ liệu đã nhập
     */
    public void write() {
        Calendar calendar = Calendar.getInstance();
        Locale vn = new Locale("vi", "VN");
        DateFormat df = DateFormat.getDateInstance(DateFormat.MEDIUM,
        DateFormat.MEDIUM, vn);
        System.out.println "[" + df.format(calendar.getTime()) + "]: " + data);
        write2File(); // Ghi dữ liệu ra file
        data = null;
    }

    private void write2File() {
        try {
            FileOutputStream fos = new FileOutputStream(filePath, true);
            data += "\n";
            fos.write(data.getBytes());
            fos.flush();
            fos.close();
        } catch (FileNotFoundException ex) {
            Logger.getLogger(DocGhiTuanTu.class.getName()).log(Level.SEVERE,
            null, ex);
        } catch (IOException ex) {
            Logger.getLogger(DocGhiTuanTu.class.getName()).log(Level.SEVERE,
            null, ex);
        }
    }
}

```

```

    }

    private class TaskDoc extends Thread {

        MayFax mFax;

        public TaskDoc(MayFax f) {
            mFax = f;
        }

        @Override
        public void run() {
            synchronized (mFax) {
                while (true) {
                    mFax.read();
                    try {
                        mFax.notify();
                        mFax.wait();
                    } catch (InterruptedException ex) {
                        System.err.println("Lỗi đọc dữ liệu: " + ex.toString());
                    }
                }
            }
        }
    }
}

```

```

    private class TaskGhi extends Thread {

        final MayFax mFax;

        public TaskGhi(MayFax f) {
            mFax = f;
        }

        @Override
        public void run() {
            synchronized (mFax) {
                while (true) {
                    mFax.write();
                    try {
                        mFax.notify();
                        mFax.wait();
                    } catch (InterruptedException ex) {
                        System.err.println("Lỗi đọc dữ liệu: " + ex.toString());
                    }
                }
            }
        }
    }
}

```

```

    }
    }
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    DocGhiTuanTu demo = new DocGhiTuanTu();
    demo.demo();
}
}

```

## Bài thực hành 2:

**Yêu cầu:** Viết chương trình Java sử dụng Thread để thực thi song song công việc:

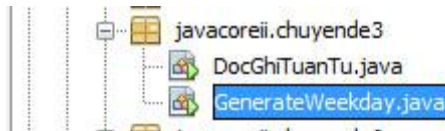
- Luồng 1: cứ mỗi giây hiển thị ngẫu nhiên một thứ trong tuần bằng tiếng Anh ("Monday", "Tuesday", "Wednesday", "Friday", "Saturday", "Thursday", "Sunday")
- Luồng 2: sau khi hiển thị thứ trong tuần bằng tiếng Anh ở luồng 1 thì luồng 2 in ra thứ tương ứng bằng tiếng Việt (vd: Sunday -> "Chủ nhật").

```

Output - JavaCorell.ChuyenDe (run)

run:
    [Thread 1]: Wednesday
[Thread 2]: Thứ tư
[Thread 2]: Thứ tư
    [Thread 1]: Wednesday
    [Thread 1]: Sunday
    [Thread 1]: Wednesday
[Thread 2]: Thứ tư
    [Thread 1]: Saturday
    [Thread 1]: Monday
[Thread 2]: Thứ hai
    [Thread 1]: Tuesday
[Thread 2]: Thứ ba
BUILD STOPPED (total time: 34 seconds)

```

**Code tham khảo:****GenerateWeekday.java**

```
package javacoreii.chuyende3;

import java.util.HashMap;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author minhvufc
 */
public class GenerateWeekday {

    String keys[] = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday", "Sunday"};
    HashMap<String, String> weekdays = new HashMap<>();

    final String T1 = "Thread 1";
    final String T2 = "Thread 2";

    String key;

    private void init() {
        weekdays.put("Monday", "Thứ hai");
        weekdays.put("Tuesday", "Thứ ba");
        weekdays.put("Wednesday", "Thứ tư");
        weekdays.put("Thursday", "Thứ năm");
        weekdays.put("Friday", "Thứ sáu");
        weekdays.put("Saturday", "Thứ bảy");
        weekdays.put("Sunday", "Chủ nhật");
    }

    // Phương thức lấy ngày tiếng Việt đồng bộ hóa
    private synchronized void getDay(String threadName) {
        if (threadName == T1) {
            Random random = new Random();
            int pos = random.nextInt(keys.length);
```

```
        key = keys[pos];
        System.out.println("\t" + "[" + threadName + "]: " + key);
    } else {
        String val = weekdays.get(key);
        System.out.println "[" + threadName + "]: " + val);
    }

    try {
        Thread.sleep(3000);
    } catch (InterruptedException ex) {
        Logger.getLogger(GenerateWeekday.class.getName()).log(Level.SEVERE,
null, ex);
    }
}

Runnable taskOne = new Runnable() {
    @Override
    public void run() {
        while (true) {
            getDay(T1);
        }
    }
};

Runnable taskTwo = new Runnable() {
    @Override
    public void run() {
        while (true) {
            getDay(T2);
        }
    }
};

private void runDemo() {
    init(); // Khởi tạo dữ liệu
    Thread t1 = new Thread(taskOne);
    Thread t2 = new Thread(taskTwo);

    t1.start();
    t2.start();
}

/**
 * @param args the command line arguments
```

```
*/  
public static void main(String[] args) {  
    GenerateWeekday gen = new GenerateWeekday();  
    gen.runDemo();  
}  
  
}
```