

Java

SỰ KIỆN

Định nghĩa sự kiện trong swing

- Là một cái gì đó thay đổi mà ta phải phản hồi lại thay đổi đấy
- Ví dụ:
 - Nhấn vào nút bấm -> đổi trạng thái
 - Hệ thống phản hồi lại việc nhấn nút
- Sự kiện có 2 phần:
 - Phần thay đổi: nhấn nút, bấm phím...
 - Phần phản hồi: luôn luôn lắng nghe (listener)
- Bản chất việc phản hồi: thực hiện code, gọi hàm chạy
- *Gọi là mô hình event - listener*

Hiểu sự kiện trong swing

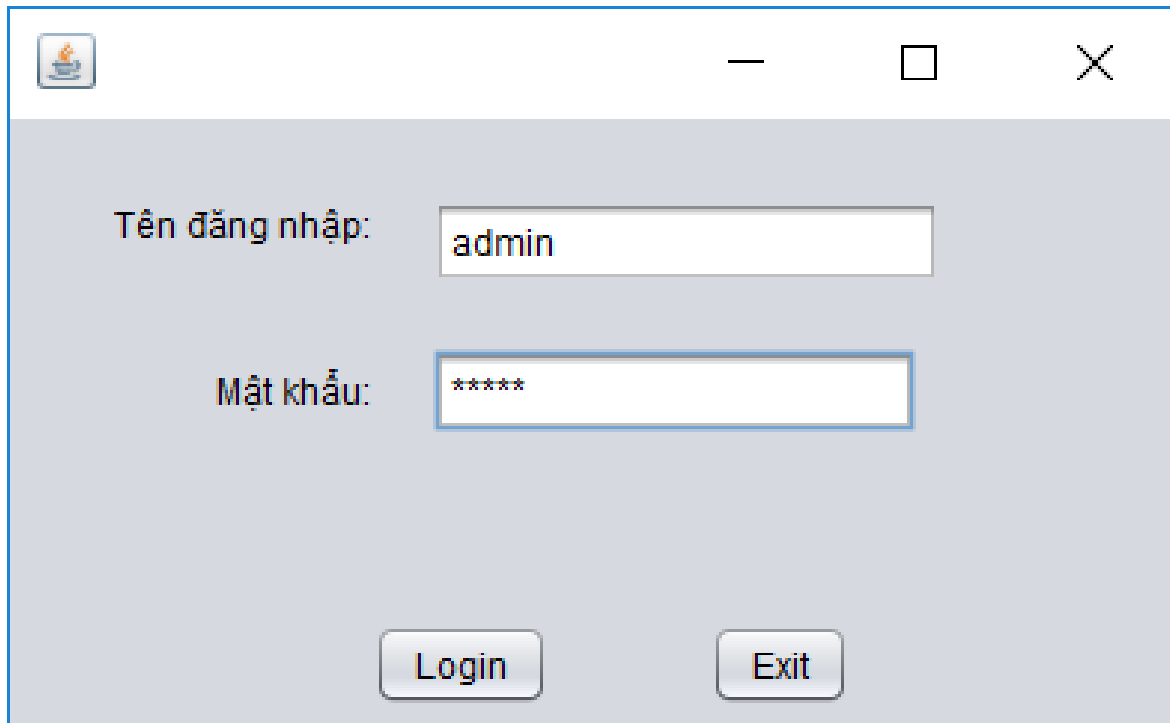
- Ai gọi hàm chạy? -> hàm thường do mình gọi
 - Hệ thống (thằng lắng nghe - listener)
 - Bao giờ gọi: không biết trc
 - Khi nào ấn thì gọi -> cần đối tượng lắng nghe
- Một nút bấm (component) có thể có nhiều giám sát (listener), một giám sát có thể giám sát nhiều nút bấm
- Vì vậy swing cung cấp rất nhiều lớp Listener
- Vậy: Nội dung hàm và giám sát (listener) có quan hệ gì không?
 - Nội dung hàm, listener do ai viết?
 - Giám sát gọi hàm do ai viết?
 - Ai có trước?

Cơ chế tổ chức sự kiện trong swing

- Vậy cơ chế gì, hay làm ntn để:
 - Nó có thể gọi đc cái hàm mà trong tương lai mới viết
 - Swing gọi đc hàm do lập trình viên viết trong tương lai
- Gọi hàm thông qua interface
 - Vậy hàm do LTV viết trong tương lai không phải là hàm tự do, mà là hàm theo khuôn dạng có sẵn
- Nên: để viết sự kiện cho nút bấm (component) mình phải:
 - Tìm hiểu xem ai lắng nghe (giám sát) việc nhấn nút (listener nào?)
 - Hay lớp nào là Listener của việc nhấn vào nút bấm

Cơ chế tổ chức sự kiện trong swing

- Hai là tìm hiểu xem, giám sát (listener) sẽ gọi hàm nào chạy mỗi khi đc nhấn nút...(hay gọi interface nào chạy)
- Demo nhanh sự kiện với Login:



The image shows a Java Swing window with a light gray background and a blue border. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main content area contains two labels: 'Tên đăng nhập:' (Username) and 'Mật khẩu:' (Password). The 'Tên đăng nhập:' label is positioned to the left of a text input field containing the text 'admin'. The 'Mật khẩu:' label is positioned to the left of a password input field containing six asterisks '*****'. At the bottom of the window, there are two buttons: 'Login' and 'Exit', both with a 3D effect and a blue border.

Tổ chức sự kiện cho chức năng login

- Kịch bản:
 - User nhấn nút login để lấy dl từ 2 textfield, kt user và pass với 2 textfield vừa nhập nếu trùng thì thông báo login thành công, else thông báo thất bại
- Như đã học: về sự kiện là chúng ta xem xét thằng nào lắng nghe, gọi hàm nào chạy, đúng chưa?
- Nhưng có một việc quan trọng hơn nữa trước khi code sự kiện?

Tổ chức sự kiện cho chức năng login

- Khi code sự kiện bước đầu tiên phải xác định xem là sự kiện của component nào?
- Hỏi: với kịch bản của login thì chúng ta cần code sự kiện của component nào?
- -> của nút bấm?
- -> component bị thay đổi thì code sự kiện cho component đấy
- Sự kiện nhấn nút là: ActionListener->là interface

Tổ chức sự kiện cho chức năng login

```
/**
```

```
 * Creates n
```

```
 */
```

```
public NewJF
```

```
    initComp
```

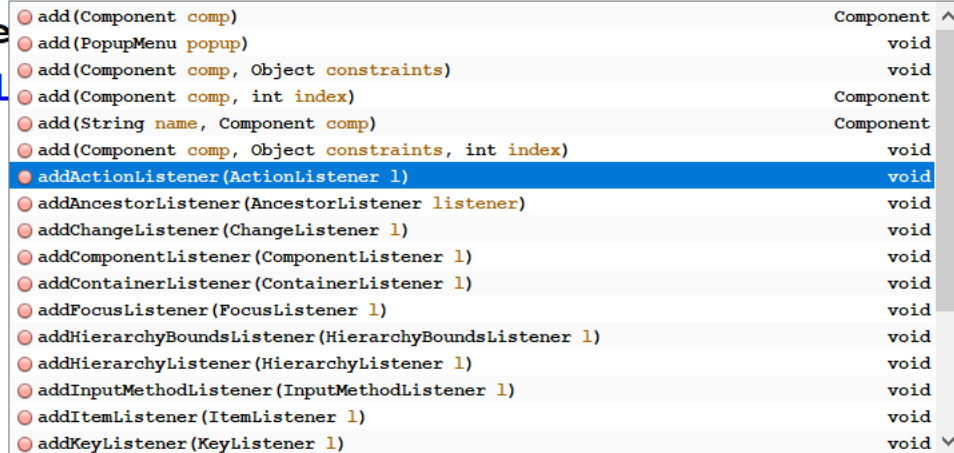
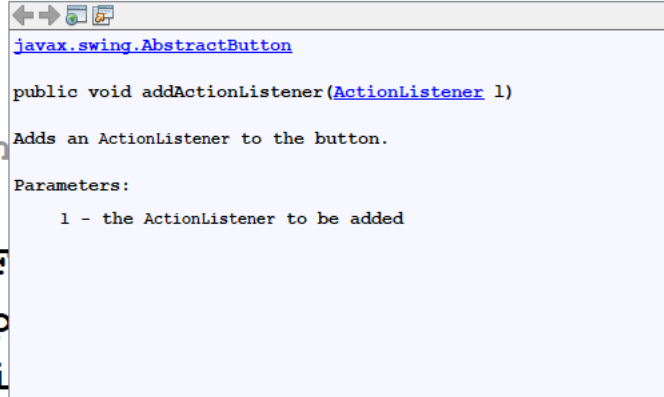
```
    Sukienki
```

```
    Sukienkichnut();
```

```
    btnLogin.addActionListener(new ActionListener() {
```

```
        @Ove
```

```
        publ
```



```
    }
```

```
}
```

```
});
```

```
{
```

```
password());
```

```
ln")){
```

```
Pane, "Đăng
```

```
Pane, "Đăng
```


Tổ chức sự kiện cho chức năng login

```
public NewJFrame() {  
    initComponents();  
    Sukienkichnut sukienkichnut = new Sukienkichnut();  
    btnLogin.addActionListener(sukienkichnut);  
    btnExit.addActionListener(new ActionListener() {  
        @Override  
        public void actionPerformed(ActionEvent e) {  
            //System.out.println("Bạn vừa nhấn nút: " + e.getSource());  
            System.exit(0);  
        }  
    });  
}
```

Tổ chức sự kiện cho chức năng login

- Yêu cầu của hàm `addActionListener` là đưa vào object của lớp `ActionListener`.
- Object của `ActionListener` có tạo được không?
- Tại sao lại đưa vào được dẫn xuất của `ActionListener`?
- Có cách nào khác không cần dẫn xuất của `ActionListener` mà vẫn có thể tạo được object của `ActionListener` để truyền vào hàm không?
- Chạy thử code chúng ta thấy mỗi lần ấn nút hàm `actionPerformed` của `ActionListener` được gọi-> mô hình sự kiện trong swing

Tổ chức sự kiện cho chức năng login

- Gắn giám sát (listener) đang giám sát nút login vào nút exit để giám sát thêm nút này được không?
- Nút login hoặc exit gắn thêm giám sát (listener) khác để giám sát sự thay đổi khác được không?
- Như vậy: 1 giám sát (listener) có thể giám sát nhiều nút bấm (component) và 1 nút bấm (component) có thể được nhiều giám sát khác nhau cùng giám sát

Tổ chức sự kiện cho chức năng login

- Đề xuất cách code sự kiện ngắn gọn hơn, súc tích hơn?
 - Sử dụng inner class (lớp trong)
 - Anonymous class (lớp vô danh)
- Code demo

Tổ chức sự kiện cho chức năng login

- Một số coder làm theo cách sau:
 - Lớp giao diện implement thẳng ActionListener
 - Trong hàm addActionPerformed truyền vào this
 - Ưu điểm: trong hàm actionPerformed lấy dữ liệu từ giao diện 1 cách dễ dàng vì cùng class
 - Nhược điểm: nếu giao diện chứa nhiều sự kiện thì sẽ rối rắm code

Tổ chức sự kiện cho chức năng login

- Đề xuất cách tối ưu (cũng là cách mà kéo thả đang làm)
 - Trong hàm `addActionPerformed` truyền vào object là đối tượng của lớp vô danh và cài đặt luôn hàm `actionPerformed`
 - Trong hàm `actionPerformed` gọi hàm xử lý sự kiện do LTV code
 - Ưu điểm: trong hàm `actionPerformed` lấy dữ liệu từ giao diện 1 cách dễ dàng vì cùng class
 - Ưu điểm: code đã được phân chia ra, tường minh hơn, dễ quản lý hơn

Tổ chức sự kiện cho chức năng login

- Co gọn lại câu chuyện tạo sự kiện
 - Sự kiện của nút bấm (component) nào?
 - Sự kiện đấy là sự kiện gì? Vì có rất nhiều sự kiện như nhấn nút, bấm phím, di chuột v.v...
 - Lớp nào giám sát (listener) sự kiện đấy.
 - ActionEvent e: chứa thông tin gì (biết cách khai thác thông tin đấy)
- Tìm hiểu thêm về slider và processbar
- Tự ngồi code sự kiện cho slider hoặc processbar
 - Kéo đúng vị trí 50 thì thông báo ra: bạn kéo đúng rồi
 - Nhả chuột ra mà không đúng vị trí thì yêu cầu kéo lại

HẾT
THỰC HÀNH