

# java GUI

---

Date: Feb. 15th, 2017

Author: xiang

## Graphical User Interface

Java的GUI部件在java.awt(抽象窗口工具包, 需调用本地系统方法实现功能, 重量级组件, 不同系统窗口的样子不同)和javax.swing(基于awt,不同系统, 组件的样子都一样, 轻量级控件, 移植性更强)两个包中。

基本组件继承关系:

```
Component (在awt里)
|--Button
|--Label
|--Checkbox 复选框
|--TextComponent 文本部件
    |--TextArea 内容 (文本区域)
    |--TextField 标题 (文本框)
|--Container 容器
    |--Panel 面板 (窗体的一部分, 不能独立存在)
    |--Window 窗体 (能独立存在)
        |--Frame 框架
        |--Dialog 对话框 (当弹出来的)
            |--FileDialog (文件选取对话框)
```

## Layout 布局

---

容器中组件的排放方式, 是布局

- FlowLayout 流式布局
  - 左到右
  - Panel的默认布局
- BorderLayout 边界布局
  - 东南西北中
  - Frame的默认布局
- GridLayout 网格布局
- CardLayout 卡片布局(选项卡)
- GridBagLayout 网格包布局

e.g.

```
public class FrameDemo {
    public static void main(String[] args) {
        // 1. 创建窗体 最初不可见的Frame对象
        Frame f = new Frame("My frame");

        //f.setSize(500, 400); //横轴，纵轴
        //f.setLocation(400, 150); // 窗体出现的位置

        // 2. 设置各种边界，位置，布局
        f.setBounds(400, 200, 500, 400); //代表上面两句

        f.setLayout(new FlowLayout()); //设置流式布局

        // 3. 创建并添加组件
        Button but = new Button("一个按钮");

        f.add(but); // 大按钮，Frame对象默认"BorderLayout"

        // 4. 设置可见
        f.setVisible(true);
        System.out.println("over");
    }
}
```

## 事件监听机制

- 事件源（组件）
- 事件（Event）
- 监听器（Listener）
- 事件处理（引发事件后处理方式）

主人公： 小强 <-----保镖 保护小强，主管小强 拳打（被拳打）{} 脚踢（被脚踢）{踢回}

事儿： 被揍（拳打，脚踢）

拳头（外力），打小强，被揍事发生在小强身上。

事件源： 小强

事件： 事儿，有些事儿在小强身上无法发生

监听器： 保镖，可能保护多人，发生了被打的事，应传到保镖这

处理方式： 拳打（被拳打）{挡住} 脚踢（被脚踢）{踢回}

# 窗体监听

**CAUTION:** 窗体监听 `addWindowListener()`

```
public class FrameDemo {
    public static void main(String[] args) {
        Frame f = new Frame("My frame");
        f.setBounds(400, 200, 500, 400);
        f.setLayout(new FlowLayout());

        Button but = new Button("一个按钮");
        f.add(but);

        //CAUTION: addWindowListener() 窗体监听
        // 请个啥样的保镖?
        f.addWindowListener(new WindowAdapter() {

            @Override
            public void windowClosing(WindowEvent e) {
                // 这很像异常处理, catch(e){}自动接收try{}中引发的异常对象
                System.out.println("closing....." + e);
                System.exit(0); // 关窗体
            }
        });

        f.setVisible(true);
        System.out.println("over");
    }
}
```

# 按钮监听

**CAUTION:** 按钮监听 `addActionListener()`

在上述代码, 窗体监听器后, 添加如下代码:

通常 `addXxxListener(new XxxListener(){{要覆盖的方法}})`

```
// 按钮上加一个监听。
but.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
});
```

# 鼠标监听

**CAUTION:** 鼠标监听 `addEventListener()`

鼠标事件：按下，释放，单击，进入 或 离开

下面代码提到的：

- 按钮同时有action和click两种监听，谁先被触发？ 答：click先触发
- 双击按钮，这事件，代码咋写？

```

/**
 * 鼠标键盘监听示例
 */
public class MouseAndKeyboardDemo {

    private Frame f;
    private TextField tf;
    private Button but;

    // alt shift s
    public MouseAndKeyboardDemo() {
        init();
    }

    private void init() {
        f = new Frame("演示鼠标和键盘监听");
        f.setBounds(400, 200, 500, 400);
        f.setLayout(new FlowLayout());

        tf = new TextField(35);
        but = new Button("一个按钮");
        f.add(tf);
        f.add(but);

        myEvent(); // 添加事件监听

        f.setVisible(true);
    }

    private void myEvent() {

        // 给Frame对象添加 “关窗体监听”
        f.addWindowListener(new WindowAdapter() {

            @Override
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });

        // 实验： 按钮同时有action和click两种监听，谁先被触发？
        // 答： click先被触发
        // but.addActionListener(new ActionListener() {
        //
        //     @Override
        //     public void actionPerformed(ActionEvent e) {
        //         System.out.println("action done");
        //     }
        // });
    }
}

```

```

// 鼠标事件：按下，释放，单击，进入 或 离开
// 要在 按钮上添加一个鼠标监听
but.addMouseListener(new MouseAdapter() {

    private int count = 1;

    @Override
    public void mouseEntered(MouseEvent e) {
        // System.out.println("我进");
        // tf.setText("我进..." + this.count++);
    }

    @Override
    public void mouseClicked(MouseEvent e) {
        // 双击按钮，这种事件，代码咋写
        if (e.getClickCount() == 2)
            tf.setText("我点..." + this.count++);
        // System.out.println("click done"+count++);
    }

});

}

public static void main(String[] args) {
    new MouseAndKeyboardDemo();
}
}

```

## 键盘监听

**CAUTION:** 键盘监听 `addKeyListener()`

在上述代码的myEvent()里添加如下代码：

下面代码展示了：

- 记录键盘按键
- 只识别键盘的数字，不识别的键，咋办？
- ctrl + enter 咋办？

```

// 给文本框添加键盘监听
// 1.确定事件源
// 2.确定事件和监听器
// 3.确定具体动作，并把要设定的内容写入该动作所示的方法体

tf.addKeyListener(new KeyAdapter() {

    @Override
    public void keyPressed(KeyEvent e) {
        // 1. 记录键盘按键
        // System.out.println("keyboard..." + e.getKeyChar() + "... " + e.getKeyCode
        ());
        // 展示shift ^^
        // System.out.println("keyboard..." +
        // KeyEvent.getKeyText(e.getKeyCode()) + "... " +
        // e.getKeyCode());
        //
        // 2. 只认数字，不认别的键，咋办？
        // int code = e.getKeyCode();
        // if(!(code>=KeyEvent.VK_0 && code<=KeyEvent.VK_9)){
        // System.out.println("必须是数字");
        // e.consume(); //不起作用
        // }

        // 3. ctrl + enter 咋办??
        if (e.isControlDown() && e.getKeyCode() == KeyEvent.VK_ENTER) {
            System.out.println("enter run...");
        }
    }
});

```

## 向Eclipse安装swing插件

<http://jingyan.baidu.com/album/4853e1e57194641909f7269f.html?picindex=1>

步骤:

1. 查看自己的Eclipse版本： 点击elcipse界面下拉菜单的 help->about eclipse
2. 百度搜索 "windowbuilder", 进入第一个搜索结果
3. 进入界面，点 "Download"
4. 选对应版本，点"Link"
5. 进入界面，复制浏览器地址栏中的地址
6. 打开Eclipse-Help-Install new Software
7. 将刚才复制的地址粘贴到work with中。并选中下面的多个选项（不同Eclipse版本，选项个数不同，都选上）

8. 点击各种next, “同意”, finish (中间大约7分钟安装下载时间)
9. 重启动Eclipse
10. 重启之后在新建里出现“WindowBuilder”说明安装成功

## 小例子

e.g.:

```
public class JFrameDemo extends JFrame {
    private JPanel contentPane;
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    JFrameDemo frame = new JFrameDemo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
    public JFrameDemo() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(new FlowLayout(FlowLayout.CENTER, 5, 5));

        JButton button = new JButton("退出");
        button.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                System.exit(0);
            }
        });
        contentPane.add(button);
    }
}
```

再来一个例子,实现 **c:\** 一回车



```

public class MyWindow extends JFrame {

    protected static final String LINE_SEPARATOR = System.getProperty("line.separator");
    private JPanel contentPane;
    private JTextField textField;
    private JTextArea textArea;
    private JButton btnNewButton;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    MyWindow frame = new MyWindow();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public MyWindow() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 597, 461);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        textField = new JTextField();
        textField.setBounds(12, 23, 392, 21);
        contentPane.add(textField);
        textField.setColumns(10);
        textArea = new JTextArea();
        textArea.setBounds(12, 95, 573, 297);
        contentPane.add(textArea);

        btnNewButton = new JButton("转到");
        btnNewButton.setBounds(439, 20, 107, 27);
        contentPane.add(btnNewButton);

        textField.addKeyListener(new KeyAdapter() {
            @Override

```

```

        public void keyPressed(KeyEvent e) {

            if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                showDir();
            }
        }
    });

    btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            showDir();
        }
    });
}

protected void showDir() {
    /*
     * 通过点击按钮，获取文本框中目录， 将目录中的内容显示在文本区域中。
     */
    String dir_str = textField.getText();
    File dir = new File(dir_str);

    if (dir.exists() && dir.isDirectory()) {
        // textArea.setText("is directory");
        textArea.setText("");
        String[] names = dir.list();
        for (String name : names) {
            textArea.append(name + LINE_SEPARATOR);
        }
    }
}
}

```

## 添加滚动条

针对上面代码，删掉 `Jpanl` 容器，取而代之的添加 `JScrollPane` 容器，在它上面添加 `JTextArea` 组件。如下：

```

public class MyWindow extends JFrame {

    protected static final String LINE_SEPARATOR = System.getProperty("line.separator");
    private JPanel contentPane;
    private JTextField textField;
    private JButton btnNewButton;
    private JTextArea textArea;

    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    MyWindow frame = new MyWindow();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    public MyWindow() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 597, 461);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        textField = new JTextField();
        textField.setBounds(12, 23, 392, 21);
        contentPane.add(textField);
        textField.setColumns(10);

        btnNewButton = new JButton("转到");
        btnNewButton.setBounds(439, 20, 107, 27);
        contentPane.add(btnNewButton);

        JScrollPane scrollPane = new JScrollPane();
        scrollPane.setBounds(12, 80, 534, 309);
        contentPane.add(scrollPane);

        textArea = new JTextArea();
        scrollPane.setViewportViewView(textArea);

        textField.addKeyListener(new KeyAdapter() {
            @Override
            public void keyPressed(KeyEvent e) {

```

```

        if (e.getKeyCode() == KeyEvent.VK_ENTER) {
            showDir();
        }
    }
});

btnNewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        showDir();
    }
});
}

protected void showDir() {
    /*
     * 通过点击按钮，获取文本框中目录， 将目录中的内容显示在文本区域中。
     */
    String dir_str = textField.getText();
    File dir = new File(dir_str);

    if (dir.exists() && dir.isDirectory()) {
        // textArea.setText("is directory");
        textArea.setText("");
        String[] names = dir.list();
        for (String name : names) {
            textArea.append(name + LINE_SEPARATOR);
        }
    }
}
}
}

```

## 添加下拉菜单 **Menu**

- JMenuBar
- 在 JMenuBar 上添加 JMenu
- 在 JMenu 上添加 JMenuItem

06 gui 14:04