Lập trình Java cơ bản

Cao Đức Thông - Trần Minh Tuấn
cdthong@ifi.edu.vn, tmtuan@ifi.edu.vn

Bài 3. Các thành phần GUI

- Một ví dụ đơn giản
- Mô hình xử lý sự kiện
- Các thành phần GUI cơ bản
- Sự kiện chuột
- Sự kiện bàn phím
- Bộ quản lý trình bày (layout)
- Bài tập

Ví dụ: Applet tính tổng 2 số

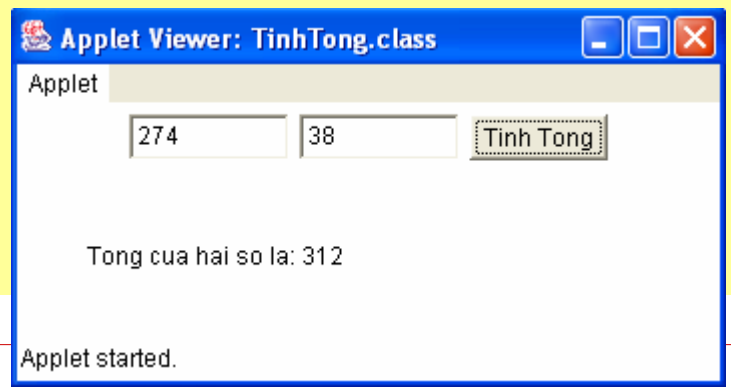
```
// file TinhTong.java
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class TinhTong extends Applet implements ActionListener
{
    private TextField txtNum1, txtNum2;
    private Button button;

    // phuong thuc nay duoc goi khi applet khoi tao
    public void init()
    {
        txtNum1 = new TextField(8); // tao o nhap so 1
        txtNum2 = new TextField(8); // tao o nhap so 2
        button = new Button("Tinh Tong"); // tao nut an
```

Ví dụ: Applet tính tổng 2 số

```
    // dua cac thanh phan vao applet
    add(txtNum1);
    add(txtNum2);
    add(button);

    // khoi tao gia tri cho o nhap
    txtNum1.setText("0");
    txtNum2.setText("0");

    // dat nghe su kien bam nut
    button.addActionListener(this);
};

// phuong thuc nay duoc goi khi co mot hanh dong xay ra
public void actionPerformed(ActionEvent event)
{
        repaint();
}
```

Ví dụ: Applet tính tổng 2 số

```
    // phuong thuc nay duoc goi khi ve lai cua so
    public void paint(Graphics g)
    {
        int num1, num2, sum;
        String s1 = txtNum1.getText();
        num1 = Integer.parseInt(s1);
        String s2 = txtNum2.getText();
        num2 = Integer.parseInt(s2);
        sum = num1 + num2;
        g.drawString("Tong cua hai so la: "+sum, 35, 80);
    }
}
```



Ví dụ: Applet tính tổng 2 số

- Giải thích applet
  - TextField và Button là các lớp thuộc gói java.awt
  - ActionListener và ActionEvent là các lớp thuộc gói java.awt.event
  - TinhTong cần cài đặt giao diện ActionListener vì nó sẽ trực tiếp xử lý sự kiện ấn nút

    button.addActionListener(this);
  - Có thể dùng một lớp khác để nghe sự kiện thay cho lớp TinhTong

Mô hình xử lý sự kiện

- Sự kiện (event) được phát sinh khi người dùng tương tác với GUI, ví dụ: di chuyển chuột, ấn nút, nhập dữ liệu văn bản, chọn menu...
- Thông tin về sự kiện được lưu trong một đối tượng thuộc lớp con của lớp AWTEvent (gói java.awt.event).
- Chương trình có thể xử lý các sự kiện bằng cách đặt "lắng nghe sự kiện" trên các thành phần GUI.

Mô hình xử lý sự kiện

- Ba thành phần chính của mô hình
  - Event source: nguồn gây ra sự kiện, thường là các thành phần GUI trong chương trình
  - Event object: đối tượng lưu thông tin về sự kiện đã xảy ra
  - Event listener: đối tượng sẽ nhận được thông tin khi có sự kiện xảy ra
  - *Event source lưu một danh sách các Event listener và sẽ thông báo cho chúng biết mỗi khi có sự kiện xảy ra*

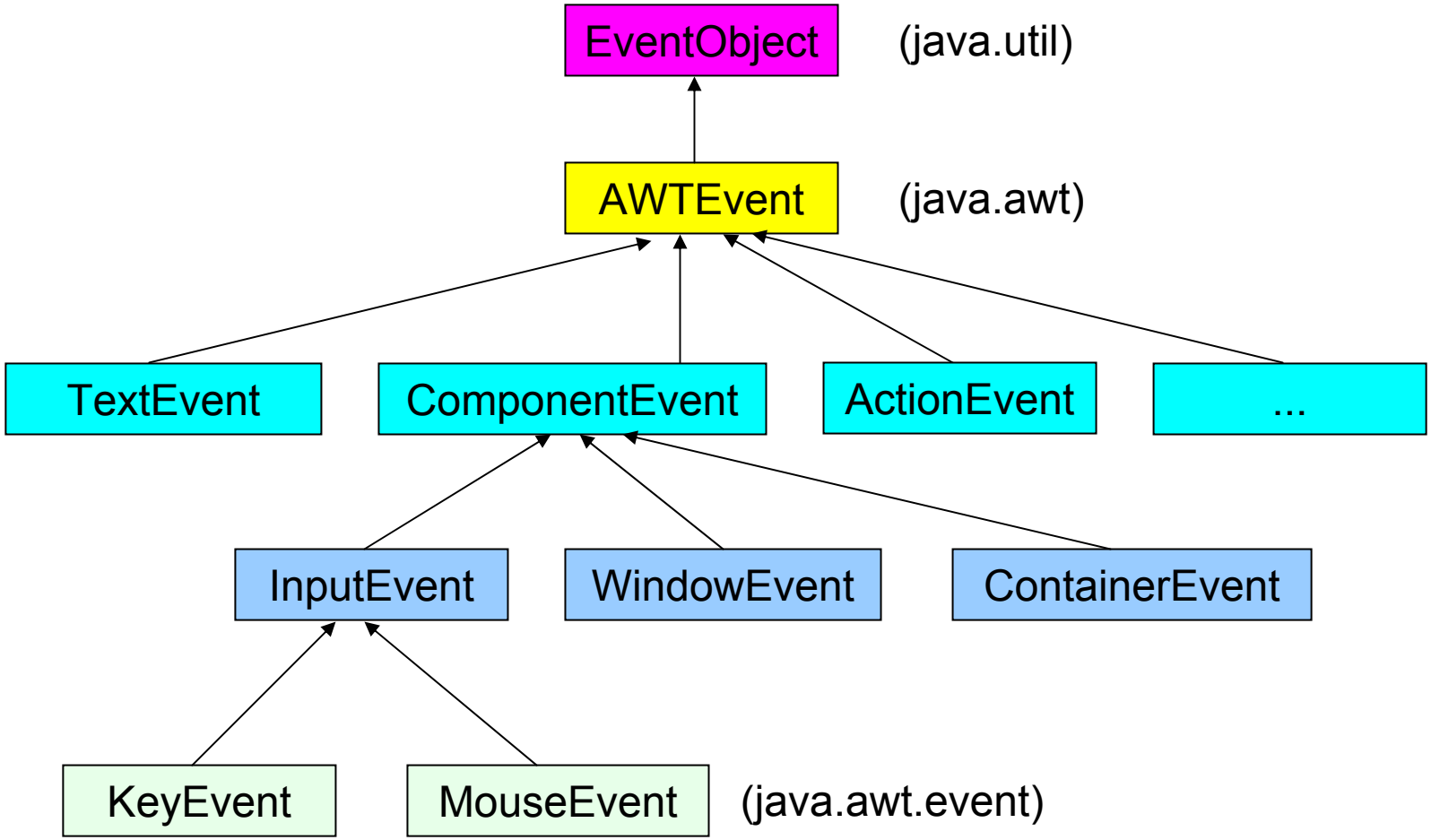
Mô hình xử lý sự kiện

- Ví dụ:

Việc thông báo sự kiện xảy ra thực chất là việc gọi một phương thức của EventListener với đối số truyền vào là EventObject.

Các lớp con của EventListener có thể cài đặt các phương thức để xử lý sự kiện.

Một số lớp sự kiện



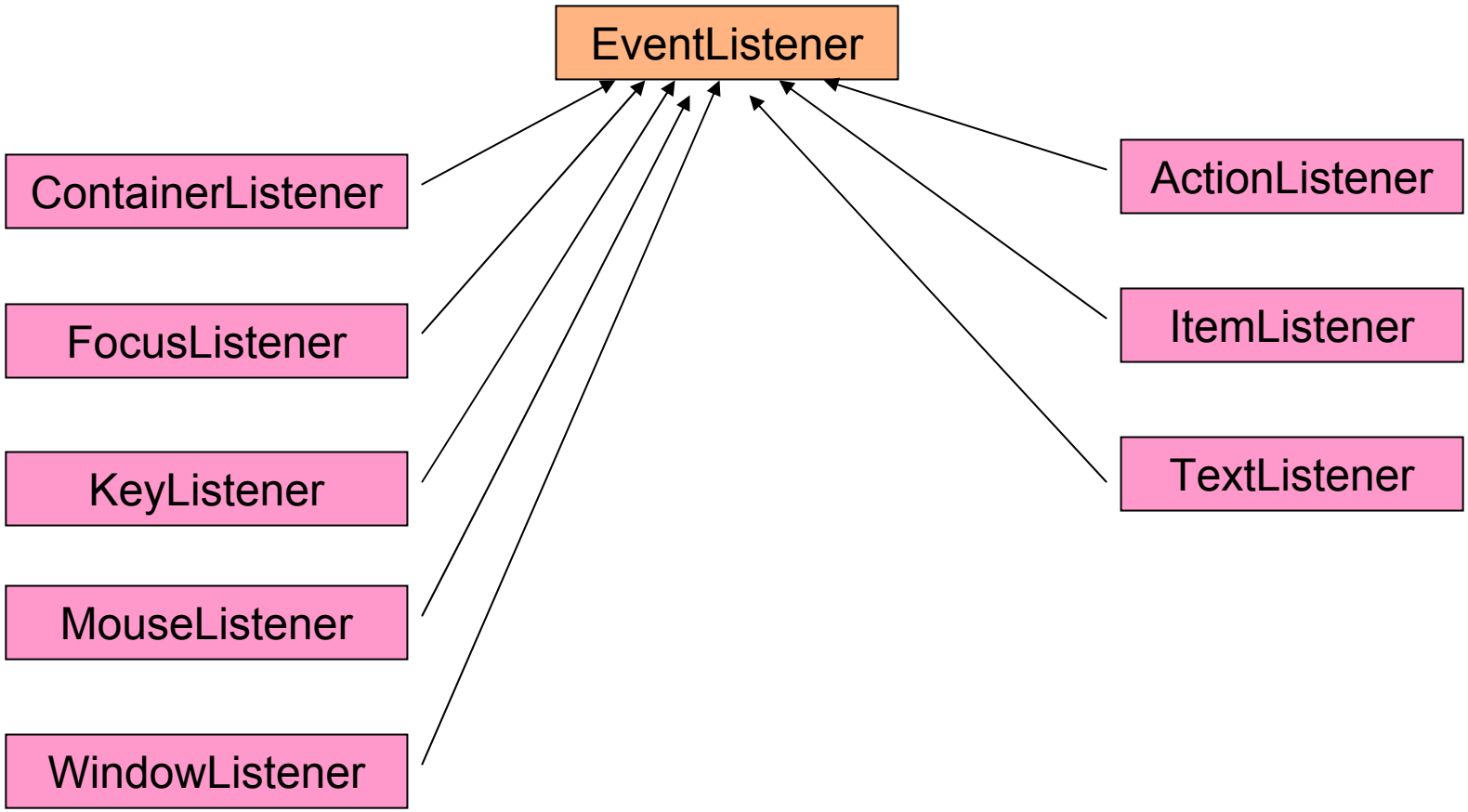
Một số lớp sự kiện

- Sự kiện cấp thấp: dùng cho hầu hết các thành phần
  - FocusEvent: đặt/chuyển focus
  - InputEvent: sự kiện phím (KeyEvent) hoặc chuột (MouseEvent)
  - ContainerEvent: thêm hoặc xoá các component
  - WindowEvent: đóng, mở, di chuyển cửa sổ
  - ...

Một số lớp sự kiện

- Sự kiện cấp cao: dùng cho một số thành phần đặc thù
  - ActionEvent: sự kiện sinh ra từ các thành phần giao tiếp với người dùng như nhấn một nút, chọn menu…
  - ItemEvent: lựa chọn một item trong danh sách
  - TextEvent: thay đổi giá trị của hộp text
  - ...

Một số interface nghe sự kiện



Cài đặt quản lý sự kiện

- Xác định đối tượng sẽ gây ra sự kiện (event source). *Ví dụ: nút bấm.*
- Xác định sự kiện cần xử lý trên đối tượng gây sự kiện. *Ví dụ: ấn nút.*
- Xác định đối tượng nghe sự kiện (event listener) và cài đặt các phương thức tương ứng. *Ví dụ: chính applet sẽ nghe sự kiện.*
- Đăng ký đối tượng nghe trên đối tượng gây ra sự kiện. *Ví dụ:* *button.addActionListener(...);*

Các event source và event object

Event source	Event	Chú thích
Button	ActionEvent	Nhấn nút
Checkbox	ItemEvent	Chọn, bỏ chọn một item
Choice	ItemEvent	Chọn, bỏ chọn một item
Component	ComponentEvent	Ẩn, hiện, di chuyển
	FocusEvent	Được chọn
	MouseEvent	Tương tác chuột
	KeyEvent	Tương tác bàn phím
Container	ContainerEvent	Thêm, bớt component
List	ActionEvent	Nhấp kép chuột một item
	ItemEvent	Chọn, bỏ chọn một item

Các event source và event object

Event source	Sự kiện	Chú thích
MenuItem	ActionEvent	Chọn một menu item
Scrollbar	AdjustmentEvent	Di chuyển thanh cuộn
TextComponent	TextEvent	Thay đổi văn bản
TextField	ActionEvent	Kết thúc thay đổi văn bản
Window	WindowEvent	Thay đổi cửa sổ

Bảng tham khảo đối tượng nghe và phương thức cần cài đặt

Event Class	Listener Interface	Listener Methods
ActionEvent	ActionListener	actionPerformed()
AdjustmentEvent	AdjustmentListener	adjustmentValueChanged()
ComponentEvent	ComponentListener	componentHidden()
		componentMoved()
		componentResized()
		componentShown()
ContainerEvent	ContainerListener	componentAdded()
		componentRemoved()
FocusEvent	FocusListener	focusGained()
		focusLost()
ItemEvent	ItemListener	itemStateChanged()

Bảng tham khảo đối tượng nghe và phương thức cần cài đặt

Event Class	Listener Interface	Listener Methods
KeyEvent	KeyListener	keyPressed()
		keyReleased()
		keyTyped()
MouseEvent	MouseListener	mouseClicked()
		mousePressed()
		mouseReleased()
	MouseMotionListener	mouseDragged()
		mouseMoved()
TextEvent	TextListener	textValueChanged()
WindowEvent	WindowListener	windowClosed()
		windowActivated()

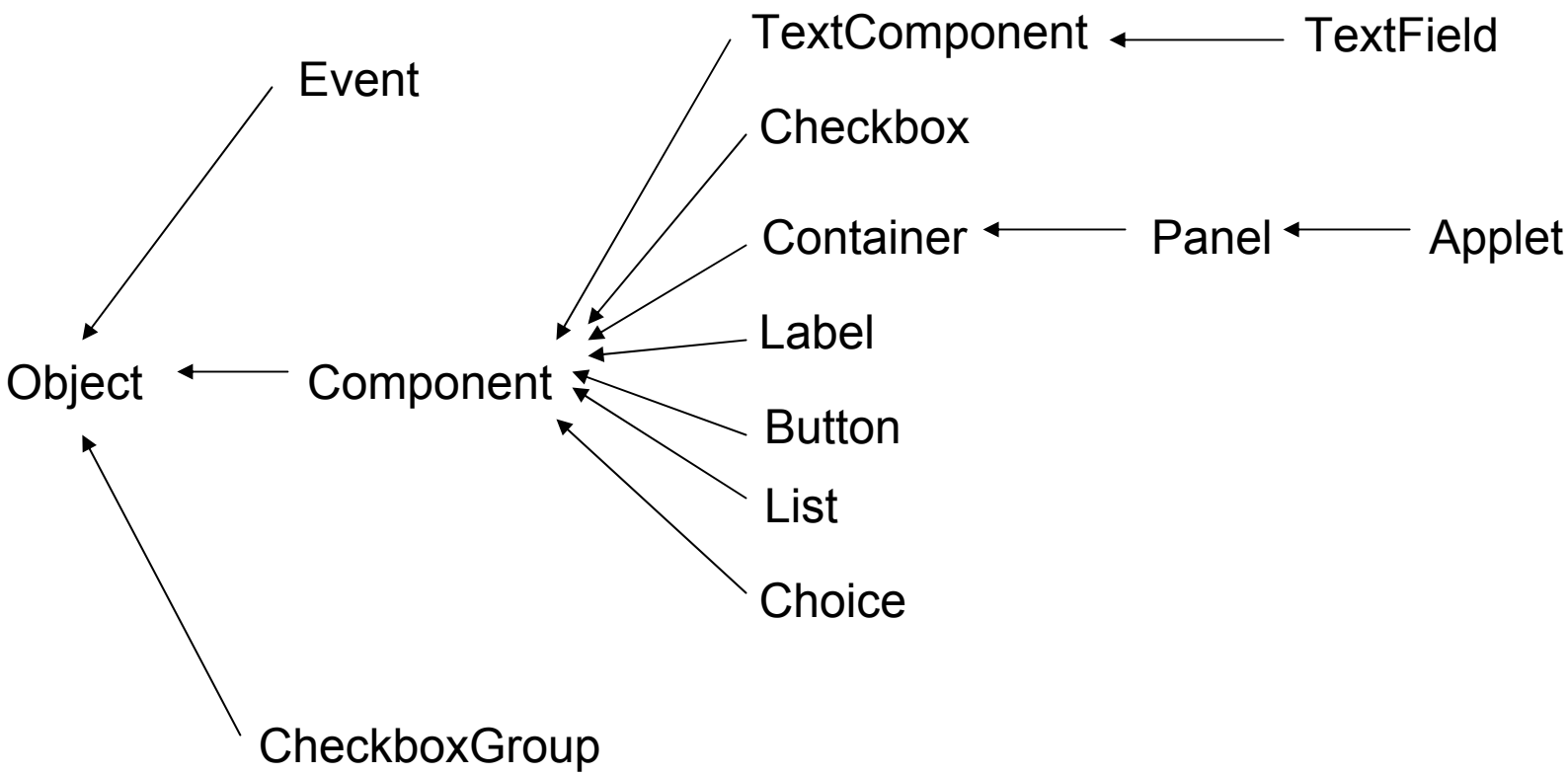
Đăng ký đối tượng nghe

- Để đăng ký đối tượng nghe ta sử dụng tên phương thức có cấu trúc như sau:

  add + loại sự kiện + Listener(lớp nghe sự kiện)

  - Ví dụ với nút Button
    - addActionListener(ActionListener)
  - Ví dụ với danh sách List
    - addActionListener(ActionListener)
    - addItemListener(ItemListener)

Một số thành phần GUI



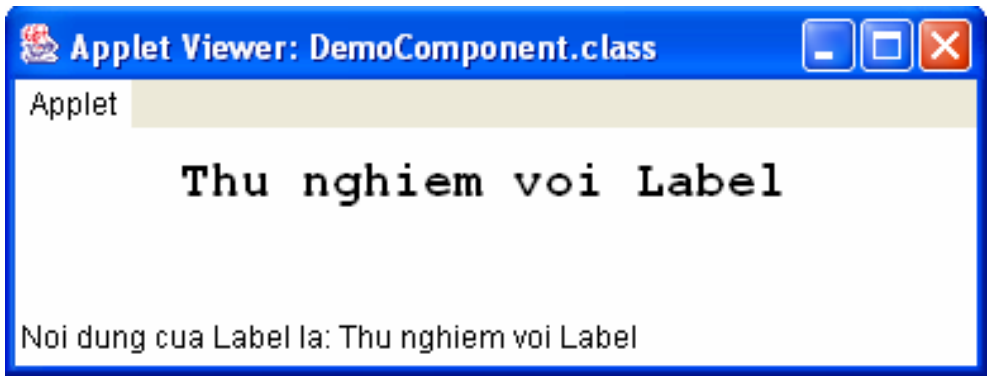
Nhãn (Label)

- Nhãn được dùng để trình bày một chuỗi văn bản ra màn hình
- Một số phương thức của Label:
  - public Label(); // tạo nhãn
  - public Label(String s); // tạo nhãn với nội dung s
  - public Label(String s, int align); // tạo và canh lề
  - void setText(String s); // đặt nội dung nhãn
  - void setAlignment(int align); // canh lề nhãn
  - ...

Nhãn (Label)

```
import java.applet.Applet;
import java.awt.*;

public class DemoLabel extends Applet
{
    private Label label;

    public void init()
    {
        Font font = new Font("Courier", Font.BOLD, 20);
        label = new Label("Thu nghiem voi Label");
        label.setFont(font);
        add(label);
     }
    public void paint(Graphics g)
    {
        showStatus("Noi dung cua Label la: " + label.getText());
    }
}
```

Nhãn (Label)


Applet Viewer: DemoComponent.class
Applet
Thu nghiem voi Label
Noi dung cua Label la: Thu nghiem voi Label

Nút nhấn (Button)

- Một số phương thức của Button
  - Button(); // tạo nút nhấn
  - Button(String s); // tạo nút nhấn có tên s
  - void setLabel(String s); // đổi tên nút
  - String getLabel(); // lấy tên nút nhấn
- Đối tượng nghe sự kiện nhấn nút cần cài đặt giao tiếp *ActionListener*

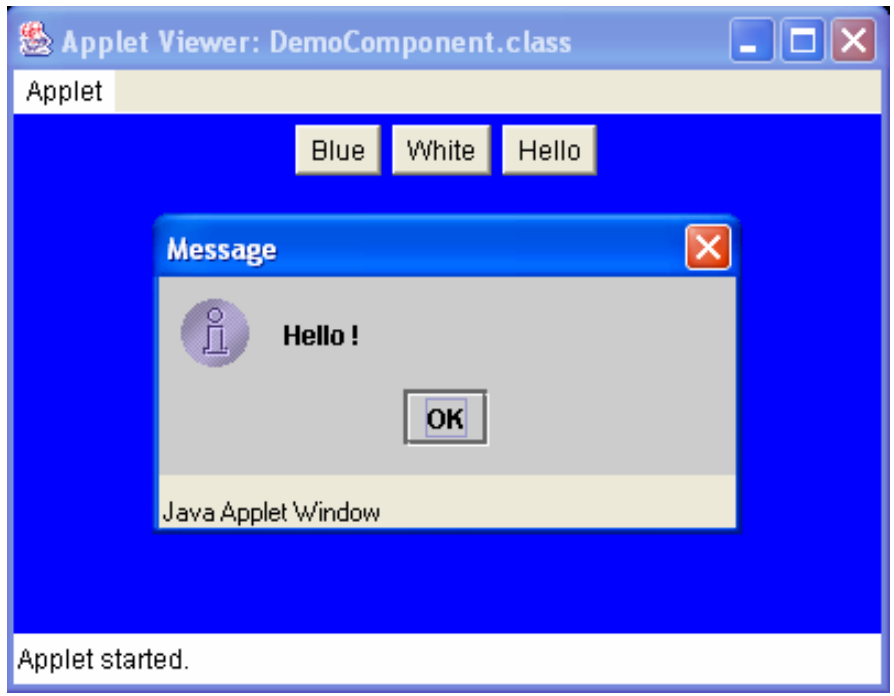
Nút nhấn (Button)

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class DemoButton extends Applet implements ActionListener
{
    private Button blueButton;
    private Button whiteButton;
    private Button helloButton;

    public void init()
    {
        blueButton = new Button("Blue");
        whiteButton = new Button("White");
        helloButton = new Button("Hello");

        blueButton.addActionListener(this);
        whiteButton.addActionListener(this);
        helloButton.addActionListener(this);
```

Nút nhấn (Button)

```
        add(blueButton);
        add(whiteButton);
        add(helloButton);
    }

    public void actionPerformed(ActionEvent event)
    {
        if (event.getSource() == helloButton)
            javax.swing.JOptionPane.showMessageDialog(this, "Hello !");
        else
        {
            if (event.getSource() == blueButton)
                this.setBackground(Color.BLUE);
            else if (event.getSource() == whiteButton)
                this.setBackground(Color.WHITE);
            repaint();
        }
    }
}
```

Nút nhấn (Button)



Ô văn bản (TextField)

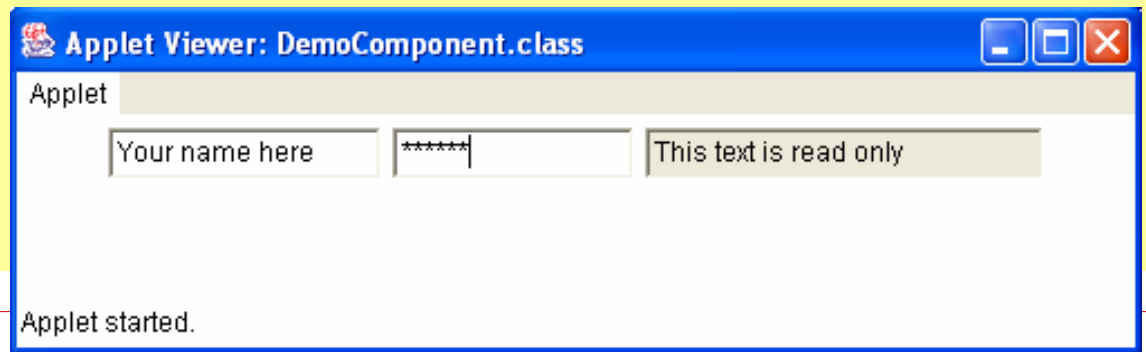
- Ô văn bản cho phép nhận dữ liệu từ bàn phím trên một dòng
- Một số phương thức
  - TextField(...); // các cấu tử
  - void setEditable(boolean b); // đặt/tắt chế độ nhập
  - void setEchoChar(char c); // đặt kí tự hiển thị
- Đối tượng nghe cần cài đặt 2 giao tiếp
  - *ActionListener*
  - *TextListener*
    - Cài đặt phương thức textValueChanged();

Ô văn bản (TextField)

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class DemoTextField extends Applet implements ActionListener
{
    private TextField txtEdit;
    private TextField txtReadOnly;
    private TextField txtPass;
    private final String PASSWORD = "Java";

    public void init()
    {
        txtEdit = new TextField("Your name here");
        txtPass = new TextField(12);
        txtPass.setEchoChar('*');
        txtPass.addActionListener(this);
        txtReadOnly = new TextField("This text is read only");
        txtReadOnly.setEditable(false);
```

Ô văn bản (TextField)

```
        add(txtEdit);
        add(txtPass);
        add(txtReadOnly);
    }

    public void actionPerformed(ActionEvent event)
    {
        if (txtPass.getText().equals(PASSWORD))
            txtReadOnly.setText("Password is valid");
        else
            txtReadOnly.setText("Invalid password !");
    }
}
```



Lựa chọn (Choice)

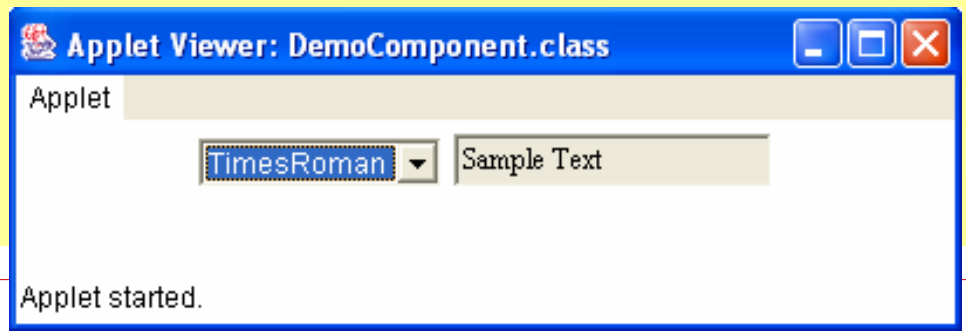
- Choice cung cấp khả năng lựa chọn một trong số các hạng mục sẵn có
- Một số phương thức
  - Choice(); // cấu tử
  - void addItem(String s); // thêm item là s
  - String getItem(int index);// lấy item có chỉ số index
  - String getSeclectedItem(); // trả về item được chọn
  - int getSelectedIndex(); // trả về index của item được chọn
- Lớp nghe cài đặt giao tiếp *ItemListener*
  - Cài đặt phương thức itemStateChanged(...)

Lựa chọn (Choice)

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class DemoChoice extends Applet implements ItemListener
{
    private Choice choice;
    private TextField txtText;
    private Font font;

    public void init()
    {
        choice = new Choice();
        choice.addItem("TimesRoman");
        choice.addItem("Courier");
        choice.addItem("Helvetica");
        choice.addItemListener(this);
```

Lựa chọn (Choice)

```
        txtText = new TextField("Sample Text", 16);
        txtText.setEditable(false);

        font = new Font(choice.getItem(0), Font.PLAIN, 12);
        txtText.setFont(font);
        add(choice);
        add(txtText);
    }

    public void itemStateChanged(ItemEvent event)
    {
        font = new Font(choice.getSelectedItem(), Font.PLAIN, 12);
        txtText.setFont(font);
    }
}
```



Checkbox (Hộp đánh dấu)

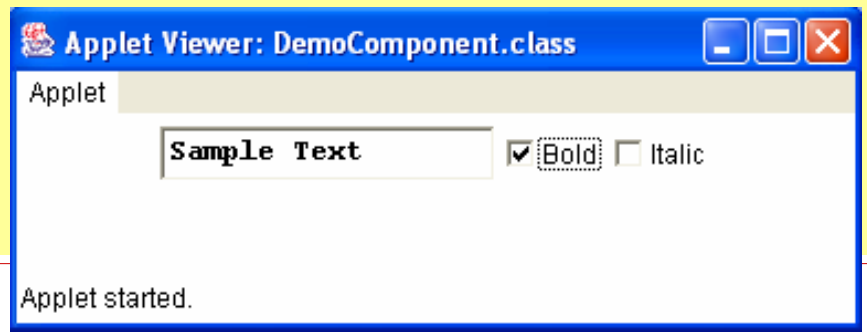
- Checkbox cung cấp các hộp tuỳ chọn cho người dùng
- Một số phương thức
  - Checkbox(...); // các cấu tử
  - void setLabel(String s); // đặt nhãn mới
  - boolean getState(); // lấy trạng thái hiện tại
- Lớp nghe cài đặt giao tiếp *ItemListener*
  - Cài đặt phương thức itemStateChanged(...)

Checkbox (Hộp đánh dấu)

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class DemoCheckbox extends Applet implements ItemListener
{
    private Checkbox checkBold;
    private Checkbox checkItalic;
    private TextField txtText;

    public void init()
    {
        checkBold = new Checkbox("Bold");
        checkItalic = new Checkbox("Italic");
        checkBold.addItemListener(this);
        checkItalic.addItemListener(this);
        txtText = new TextField("Sample Text", 16);
        Font font = new Font("Courier", Font.PLAIN, 14);
        txtText.setFont(font);
```

Checkbox (Hộp đánh dấu)

```
        add(txtText);
        add(checkBold);
        add(checkItalic);
    }

    public void itemStateChanged(ItemEvent event)
    {
        int valBold = Font.PLAIN;
        int valItalic = Font.PLAIN;
        if (checkBold.getState())  valBold = Font.BOLD;
        if (checkItalic.getState()) valItalic = Font.ITALIC;
        Font font = new Font("Courier", valBold + valItalic, 14);
        txtText.setFont(font);
    }
}
```



Checkbox và CheckboxGroup

- Các Checkbox có thể được đặt trong một CheckboxGroup để tạo ra các radio button.
- Ví dụ: Tạo 3 radio button

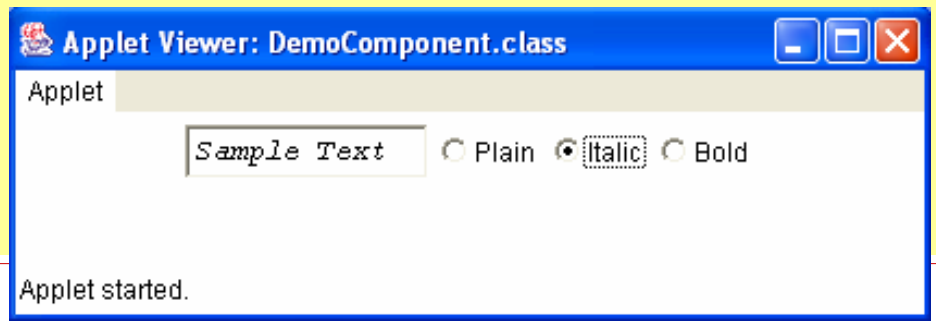
```
// Tạo 3 radio button thuộc cùng một nhóm. Ban đầu
// radio1 được chọn. Tại mỗi thời điểm chỉ có thể chọn một
// trong 3 radio.
CheckboxGroup g = new CheckboxGroup();
Checkbox radio1 = new Checkbox("Radio1", g, true);
Checkbox radio2 = new Checkbox("Radio2", g, false);
Checkbox radio3 = new Checkbox("Radio3", g, false);
```

Checkbox và CheckboxGroup

```
// Cac import can thiet...
public class DemoRadio extends Applet implements ItemListener
{
    private Checkbox plain, bold, italic;
    private CheckboxGroup group;
    private TextField txtText;

    public void init()
    {
        group = new CheckboxGroup();
        plain = new Checkbox("Plain", group, true);
        bold = new Checkbox("Bold", group, false);
        italic = new Checkbox("Italic", group, false);

        txtText = new TextField("Sample Text");
        txtText.setFont(new Font("Courier", Font.PLAIN, 14));
        plain.addItemListener(this);
        bold.addItemListener(this);
        italic.addItemListener(this);
```

Checkbox và CheckboxGroup

```
        add(txtText);
        add(plain);
        add(italic);
        add(bold);
    }

    public void itemStateChanged(ItemEvent event)
    {
        int mode = 0;
        if (event.getSource() == plain)  mode = Font.PLAIN;
        if (event.getSource() == italic)  mode = Font.ITALIC;
        if (event.getSource() == bold)   mode = Font.BOLD;
        txtText.setFont(new Font("Courier", mode, 14));
    }
}
```



Danh sách (List)

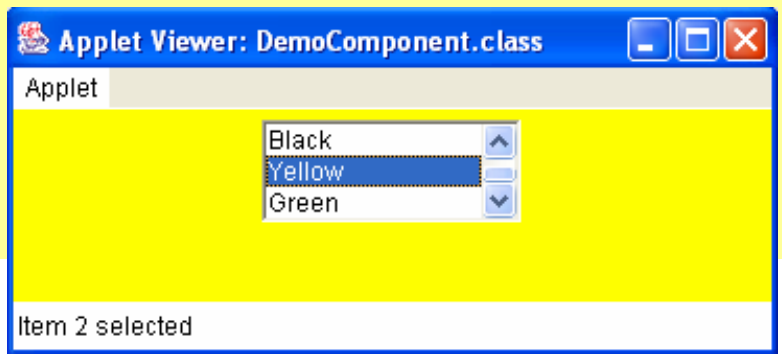
- List cho phép người dùng chọn một hay nhiều item từ một danh sách các item
- Một số phương thức
  - List(); // cấu tử mặc định
  - List(int items, boolean ms); // cấu tử mở rộng
  - String getSeclectedItem(); // lấy lại thành phần được chọn
- Lớp nghe cài đặt giao tiếp *ItemListener* và/hoặc *ActionListener*

Danh sách (List)

```
// Cac import can thiet...
public class DemoList extends Applet implements ItemListener,
                                                ActionListener
{
    private List colorList;

    public void init()
    {
        colorList = new List(3, false);
        colorList.add("White");
        colorList.add("Black");
        colorList.add("Yellow");
        colorList.add("Green");
        colorList.addItemListener(this);
        colorList.addActionListener(this);
        add(colorList);
    }
```

Danh sách (List)

```
    public void itemStateChanged(ItemEvent event)
    {
        List list = (List) event.getSource();
        showStatus("Item " + list.getSelectedIndex() + " selected");
    }
    public void actionPerformed(ActionEvent event)
    {
        List list = (List) event.getSource();
        String s = list.getSelectedItem();
        if (s.equals("White"))  setBackground(Color.WHITE);
        if (s.equals("Black"))   setBackground(Color.BLACK);
        if (s.equals("Yellow")) setBackground(Color.YELLOW);
        if (s.equals("Green"))  setBackground(Color.GREEN);
        repaint();
     }
}
```



Các sự kiện chuột

- Để quản lý các sự kiện chuột cần cài đặt giao tiếp
  - *MouseListener*
  - *MouseMotionListener*
- Các phương thức của *MouseListener*
  - void mousePressed(MouseEvent e);
  - void mouseClicked(MouseEvent e);
  - void mouseReleased(MouseEvent e);
  - void mouseEntered(MouseEvent e);
  - void mouseExited(MouseEvent e);

Các sự kiện chuột

- Các phương thức của *MouseMotionListener*
  - void mouseDragged(MouseEvent e);
  - void mouseMoved(MouseEvent e);
- Đối tượng MouseEvent
  - Chứa các thông tin về sự kiện chuột
- Ví dụ: Chương trình vẽ đơn giản

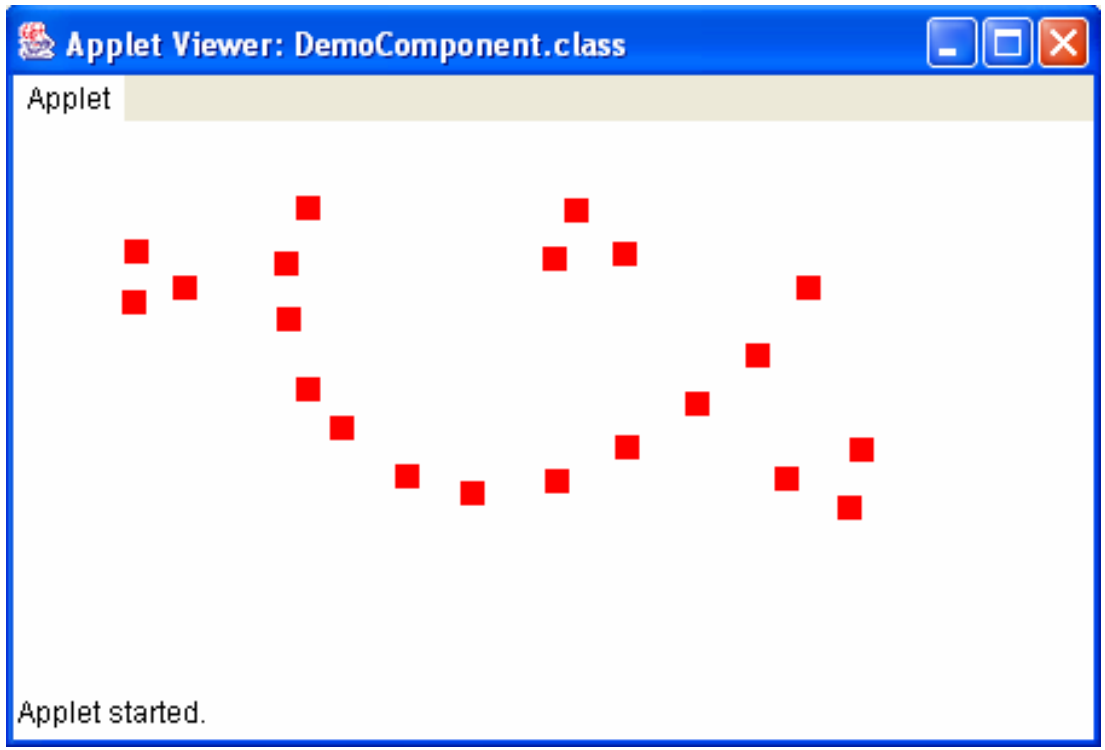
Các sự kiện chuột

```
// Cac import can thiet...
public class DemoMouse extends Applet implements MouseListener
{
    private Rectangle[] rects;
    private final int MAX_RECT = 100;
    private int numRects;

    public void init()
    {
        rects = new Rectangle[MAX_RECT];
        numRects = 0;
        addMouseListener(this);
        setForeground(Color.RED);
    }

    public void paint(Graphics g)
    {
        for(int i=0; i< numRects; i++)
            g.fillRect(rects[i].x, rects[i].y, rects[i].width, rects[i].height);
    }
```

Các sự kiện chuột

```
  public void mouseClicked(MouseEvent e)
  {
    if (numRects < MAX_RECT)
    {
      rects[numRects++]=new Rectangle(e.getX(), e.getY(), 10, 10);
      repaint();
    }
  }

   // Can cai dat tat ca cac phuong thuc cua giao tiep
   public void mousePressed(MouseEvent e) {}
   public void mouseReleased(MouseEvent e) {}
   public void mouseEntered(MouseEvent e) {}
   public void mouseExited(MouseEvent e) {}
}
```

Các sự kiện chuột



Các lớp adapter

- Khi dùng giao tiếp MouseListener ta phải cài đặt tất cả các phương thức của nó, ngay cả khi ta chỉ dùng một trong số đó.
- Java cung cấp một số lớp đã cài đặt sẵn những phương thức này gọi là các lớp Adapter). Ta chỉ cần thừa kế, cài đặt phương thức cần thiết. Các lớp adapter cũng nằm trong gói java.awt.event

Các lớp adapter

- Một số lớp adapter

Interface	Adapter class
ComponentListener	ComponentAdapter
ContainerListener	ContainerAdapter
FocusListener	FocusAdapter
KeyListener	KeyAdapter
MouseListener	MouseAdapter
MouseMotionListener	MouseMotionAdapter
WindowListener	WindowAdapter

Các sự kiện bàn phím

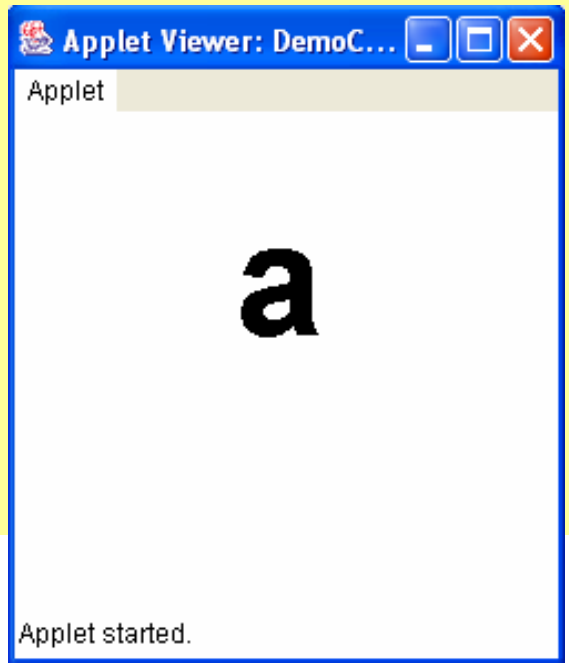
- Một lớp muốn nghe sự kiện bàn phím phải cài đặt giao tiếp *KeyListener*
  - void keyTyped(KeyEvent e);
  - void keyPressed(KeyEvent e);
  - void keyReleased(KeyEvent e);
- *Chú ý: Có thể sử dụng KeyAdapter thay cho dùng giao tiếp KeyListener*

Các sự kiện bàn phím

```
// Cac import can thiet...
public class DemoKey extends Applet implements KeyListener
{
    private String key;

    public void init()
    {
        addKeyListener(this);
        key = "";
    }

    public void paint(Graphics g)
    {
        g.setFont(new Font("Arial", Font.BOLD, 72));
        g.drawString(key, 100, 100);
    }
```

Các sự kiện bàn phím

```
    public void keyTyped(KeyEvent e)
    {
        key = "" + e.getKeyChar();
        repaint();
    }

    public void keyPressed(KeyEvent e) {}
    public void keyReleased(KeyEvent e){}
}
```



Bài tập tại lớp

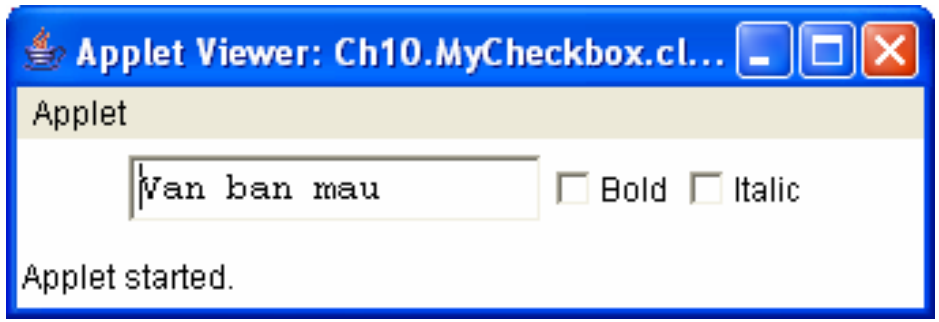
- Bài 1: Viết một applet thực hiện công việc sau: khi chuột được di chuyển vào applet thì thông báo *Hello Mouse*, khi ra khỏi applet thì thông báo *Goodbye Mouse*
- Bài 2: Viết một applet cho phép vẽ đường thẳng bằng chuột (giống MS Paint)

Bộ quản lý bố cục (Layout manager)

- Java cung cấp sẵn các lớp hỗ trợ trình bày các thành phần GUI.
- Một số lớp bố cục đơn giản
  - FlowLayout: sắp xếp tuần tự
  - BorderLayout: sắp xếp theo năm khu vực
  - GridLayout: sắp xếp theo hàng và cột
- Chú ý:
  - Với Applet và Panel, bố cục mặc định là FlowLayout. Có thể thay đổi bố cục bằng hàm setLayout

Lớp FlowLayout

- Các thành phần được đưa vào từ trái sang phải, theo từng dòng, nếu hết dòng sẽ sang dòng mới.



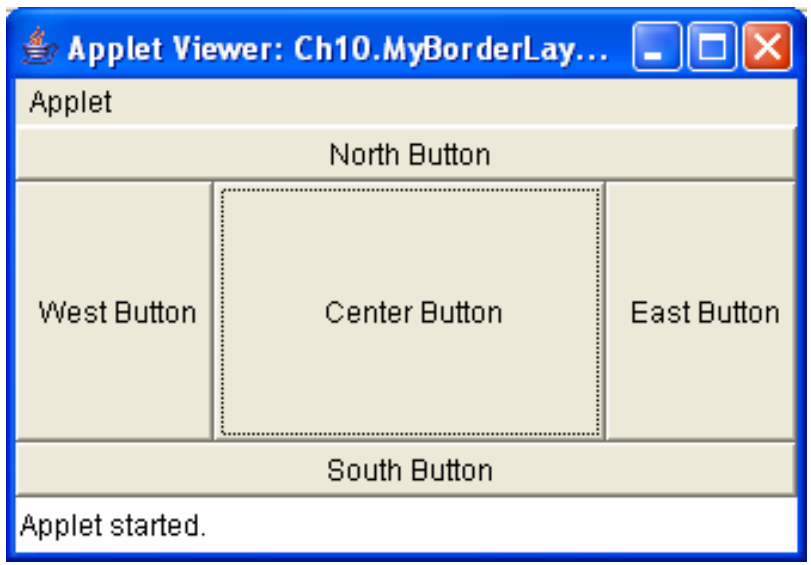
Lớp FlowLayout

- Một số phương thức của FlowLayout
  - FlowLayout(...); // các cấu tử
  - void setAlignment(int align); // căn lề

```
public void init()
{
        // tao flow layout can le phai
        FlowLayout layout = new FlowLayout(FlowLayout.RIGHT);
        setLayout(layout);
        add(new TextField(15));
        add(new Button("Press me"));
}
```

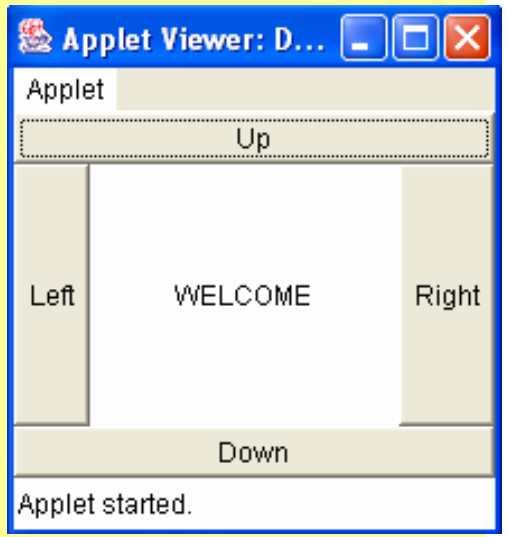
Lớp BorderLayout

- BorderLayout sắp xếp các thành phần theo 5 vùng: EAST, WEST, SOUTH, NORTH, CENTER



Lớp BorderLayout

```
// ...
public void init()
{
    // tao border layout
    setLayout(new BorderLayout());
    add(new Button("Up"), BorderLayout.NORTH);
    add(new Button("Left"), BorderLayout.WEST);
    add(new Button("Right"), BorderLayout.EAST);
    add(new Button("Down"), BorderLayout.SOUTH);
    add(new Label("WELCOME", Label.CENTER),
                                BorderLayout.CENTER);
}
```

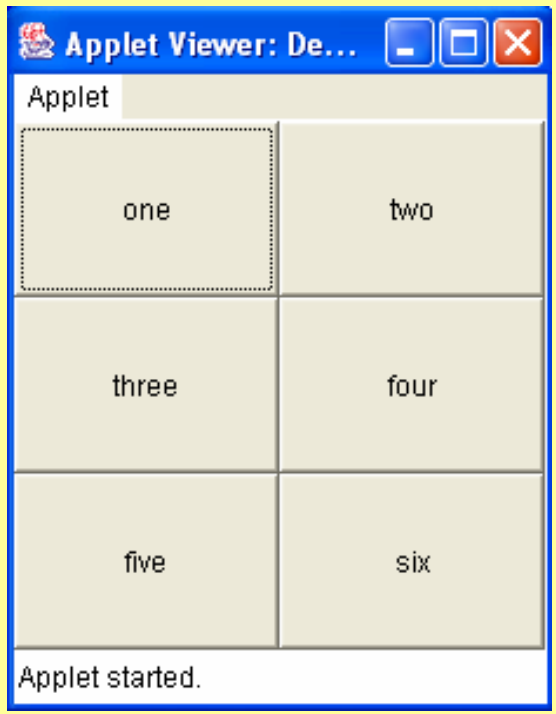


- Chú ý: Khi add một component theo BorderLayout cần chỉ rõ vùng, nếu không component sẽ không được hiển thị.

Lớp GridLayout

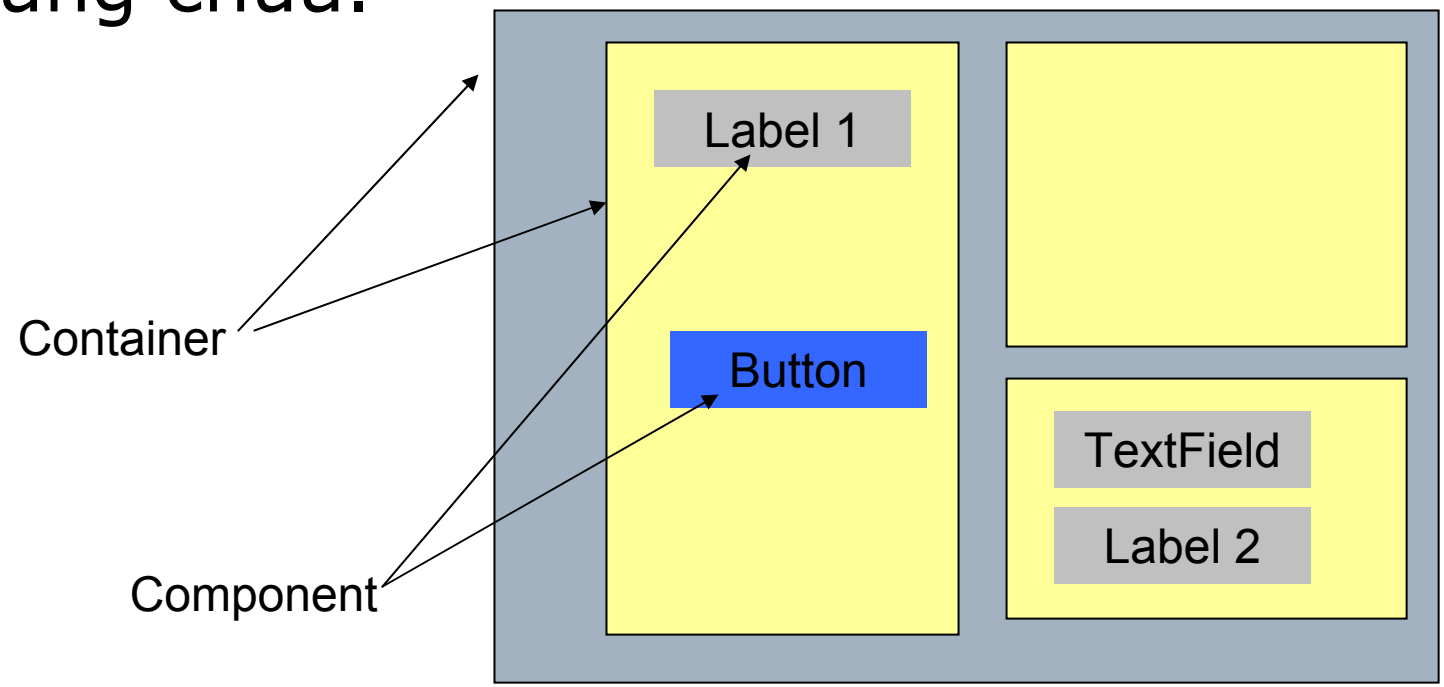
- GridLayout sắp xếp các thành phần trong một lưới có hàng và cột. Kích thước các component trong GridLayout là như nhau.

```
private Button[] b;
public void init()
{
   // tao grid layout
   b = new Button[6];
   b[0] = new Button("one");
   b[1] = new Button("two");
   b[2] = new Button("three");
   b[3] = new Button("four");
   b[4] = new Button("five");
   b[5] = new Button("six");
   setLayout( new GridLayout(3,2) );
   for(int i=0; i<b.length; i++)  add(b[i]);
}
```



Khung chứa (Container)

- Khung chứa là các đối tượng trên đó có thể chứa các thành phần khác. Applet, Frame, Dialog, Panel là các ví dụ về khung chứa.

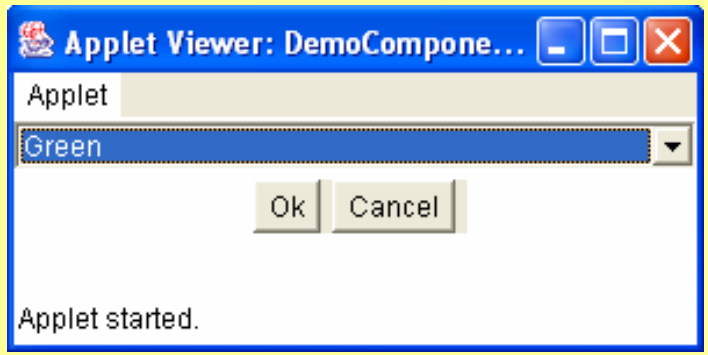


Lớp Panel (Vùng chứa)

- Lớp Panel kế thừa từ Container. Nó có thể được dùng để tạo ra các giao diện theo ý muốn.
- Ví dụ: Một giao diện có thể có nhiều panel sắp xếp theo một layout nhất định, mỗi panel lại có các component sắp xếp theo một layout riêng.
- *Chú ý: Panel có bố cục mặc định là FlowLayout.*

Lớp Panel (Vùng chứa)

```
public void init()
{
    Choice choice = new Choice();
    choice.add("Red");
    choice.add("Green");
    choice.add("Blue");

    Button ok = new Button("Ok");
    Button cancel = new Button("Cancel");
    Panel panel = new Panel();
    panel.add(ok);
    panel.add(cancel);

    this.setLayout(new BorderLayout());
    this.add(choice, BorderLayout.NORTH);
    this.add(panel, BorderLayout.CENTER);
}
```

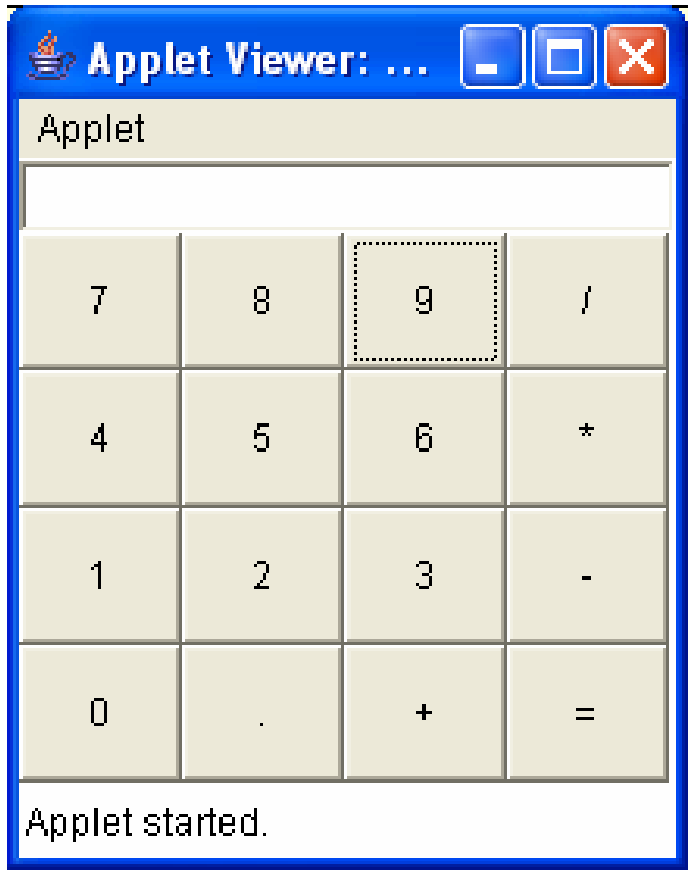


Bài tập

1. Viết applet cho phép nhập 3 hệ số a, b, c (bằng TextField) sau đó giải phương trình $ax^2 + bx + c = 0$ và in kết quả.
2. Viết applet cho phép nhập 3 hệ số a, b, c và 2 cận x1, x2 sau đó vẽ đồ thị hàm số $ax^2 + bx + c = 0$ trong đoạn [x1, x2].
3. Viết applet tạo 3 nút bấm. Bấm nút 1 thì vẽ ngẫu nhiên 100 đường thẳng ra màn hình. Bấm nút 2 thì vẽ ngẫu nhiên 50 hình tròn. Bấm nút 3 thì vẽ ngẫu nhiên 50 hình vuông.

Bài tập

4. Viết một applet cho phép tính toán đơn giản như hình bên. Chú ý người dùng có thể nhập dữ liệu bằng phím hoặc chuột.



Bài tập

5. Viết applet cho phép người dùng vẽ các hình chữ nhật bằng chuột. Khi người dùng ấn chuột, kéo và sau đó thả chuột thì một hình chữ nhật tương ứng sẽ được vẽ ra.

   Mở rộng chương trình: tạo 4 radio cho phép chọn vẽ Oval, Rectangle, Fill Oval, Fill Rectangle.

6. Viết chương trình cho phép người dùng điều khiển một quả bóng. Trên màn hình có các nút là: To, Nhỏ, Trái, Phải, Lên, Xuống. Khi người dùng ấn 1 nút thì kích cỡ/vị trí của quả bóng sẽ thay đổi theo. Yêu cầu tạo một lớp Ball riêng biệt.

Bài tập

7. Viết chương trình mô tả trò chơi dò mìn. Trên màn hình có 3x3 nút bấm và mỗi nút có thể là có mìn hoặc không (ngẫu nhiên). Khi người dùng nhấn một nút, nếu nút đó không có mìn thì cho phép người dùng ấn tiếp, còn không thì thông báo "mìn nổ" và dừng lại. Lưu ý là mỗi nút có một số và người dùng có thể nhấn phím số tương ứng thay vì nhấn chuột vào nút.