Chương 5

Xây dựng ứng dụng AWT đơn giản

5.0 Dẫn nhập
5.1 Tổng quát về lập trình giao diện đồ họa trong Java
5.2 Các phần tử giao diện trong thư viện AWT
5.3 Lập trình tạo cửa sổ giao diện (Frame)
5.4 Cơ chế bố trí các phần tử trong container
5.5 Tương tác giữa người dùng và đối tượng giao diện
5.6 Thí dụ viết phần mềm giải phương trình bậc 2
5.7 Kết chương

5.0 Dẫn nhập

- Chương này sẽ trình bày các đối tượng giao diện đồ họa được cung cấp trong thư viện AWT (Abstract Windowing Toolkit), tính chất của từng đối tượng, mối quan hệ giữa chúng.
- Chương này cũng giới thiệu qui trình lập trình tạo đối tượng Frame miêu tả cửa sổ giao diện cùng các đối tượng giao diện AWT được chứa bên trong nó theo yêu cầu của chương trình. Cuối chương sẽ trình bày mã nguồn của chương trình giải phương trình bậc 2 dùng các đối tượng giao diện AWT.

5.1 Tổng quát về lập trình giao diện đồ họa trong Java

- Trong chương 3 chúng ta đã trình bày cách viết 1 ứng dụng Java chạy ở chế độ "text-mode". Chương trình chạy ở chế độ "text-mode" chỉ có thể giao tiếp dữ liệu với người dùng theo cách thức rất cứng nhắc :
  - xuất tuần tự dữ liệu bên trong chương trình ra màn hình dưới dạng văn bản thô (không có định dạng), hết dữ liệu này đến dữ liệu khác.
  - chờ nhận tuần tự từng dữ liệu mà người dùng nhập vào (ở dạng văn bản thô).
- Cách tương tác với người dùng ở mức độ này còn khá hạn chế, không thân thiện và không tạo được độ dễ dàng sử dụng phần mềm.

5.1 Tổng quát về lập trình giao diện đồ họa trong Java

- Để người dùng dễ dàng, thích thú trong việc dùng chương trình, chương trình nên dùng giao diện đồ họa và tương tác trực quan hơn. Cụ thể là chương trình có thể :
  - nhận đồng thời nhiều dữ liệu do người dùng nhập vào, thứ tự nhập vào là không quan trọng, người dùng có thể nhập dữ liệu nào trước, dữ liệu nào sau cũng được. Dữ liệu nhập vào không chỉ có văn bản thô mà có thể là chọn lựa 1 mục trong danh sách có sẵn,...
  - xuất đồng thời nhiều dữ liệu ra màn hình, thứ tự xuất ra là không quan trọng. Hơn nữa kết quả xuất không chỉ có 1 dạng văn bản thô mà còn có thể theo định dạng bất kỳ, ngoài ra dữ liệu xuất có thể là hình bitmap, các hình đồ họa toán học như đoạn thẳng, hình tứ giác, hình cong tròn, ellipse,...

5.1 Tổng quát về thư viện AWT

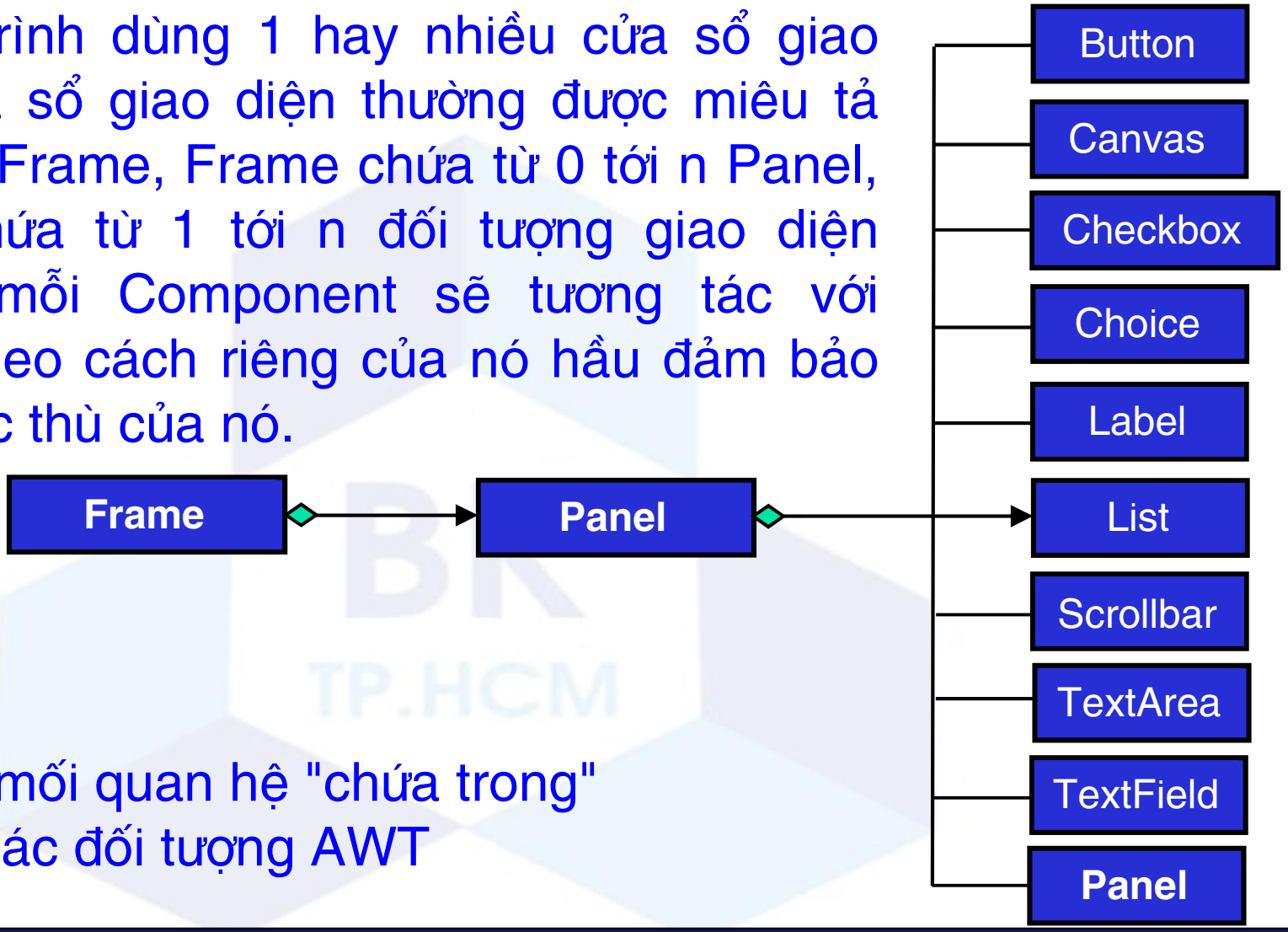
- Thư viện lập trình đồ họa cơ bản nhất và có thể chạy trên nhiều cấu hình nhất mà JDK cung cấp có tên là AWT (Abstract Windowing Toolkit). Đây là package cung cấp các đối tượng phục vụ hiển thị và xử lý các phần tử đồ họa trực quan.
- Trước hết chúng ta hãy tìm hiểu các loại phần tử chức năng mà AWT cung cấp, tính chất của chúng và mối quan hệ giữa chúng.

5.2 Các phần tử giao diện trong thư viện AWT

- Các phần tử tương tác trực tiếp với người dùng được gọi là Component, thí dụ như Button, TextField, Label,... Trong AWT, các component đều được đặc tả bởi class Component hay 9 class con của nó (Button, Canvas, Checkbox, Choice, Label, List, Scrollbar, TextArea, TextField).
- Các component không thể tồn tại 1 mình độc lập, chúng phải được chứa trong phần tử lớn hơn được gọi là Container. Container chứa và bố trí các component theo 1 cách xác định. Bản thân Container cũng là Component, do đó ta có thể chứa nó trong Container khác theo cơ chế lồng nhau nhiều cấp. Trong AWT, các Container đều được đặc tả bởi class Container hay 1 class con nào đó của nó. AWT cung cấp 4 class Container khác nhau : Window, Frame, Dialog, Panel.

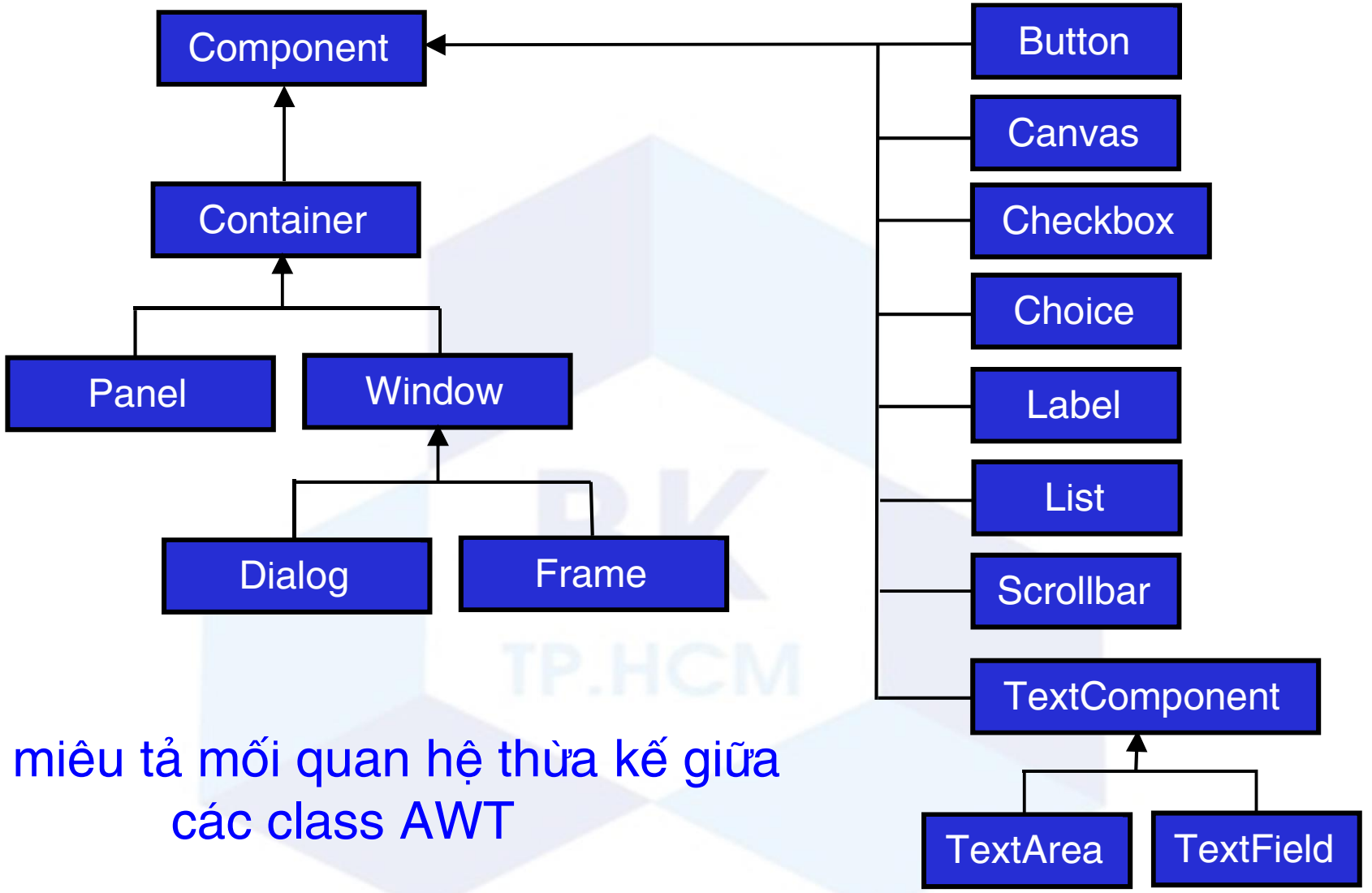
5.2 Các phần tử giao diện trong thư viện AWT

Mỗi chương trình dùng 1 hay nhiều cửa sổ giao diện, mỗi cửa sổ giao diện thường được miêu tả bởi đối tượng Frame, Frame chứa từ 0 tới n Panel, mỗi Panel chứa từ 1 tới n đối tượng giao diện Component, mỗi Component sẽ tương tác với người dùng theo cách riêng của nó hầu đảm bảo chức năng đặc thù của nó.



Cây miêu tả mối quan hệ "chứa trong" giữa các đối tượng AWT

5.2 Các phần tử giao diện trong thư viện AWT



Cây miêu tả mối quan hệ thừa kế giữa các class AWT

5.3 Lập trình tạo cửa sổ giao diện (Frame)

- Qui trình lập trình để tạo và hiển thị 1 cửa sổ giao diện chứa nhiều đối tượng giao diện bên trong :

1. tạo frame cần dùng và đối tượng quản lý layout cho nó :
   MyFrame fr = new MyFrame(); //class MyFrame được định nghĩa theo yêu cầu riêng.
   fr.setLayout(new FlowLayout());
2. tạo 1 Panel cần dùng và đối tượng quản lý layout cho nó :
   Panel mPanel = new Panel();
   mPanel.setLayout(new GridLayout());
3. tạo từng Component cần dùng và add nó vào Panel :
   mButton = new Button("OK"); mPanel.add(mButton);
   mTextField = new TextField(""); mPanel.add(mTextField);
   ...

5.3 Lập trình tạo cửa sổ giao diện (Frame)

4. add Panel vừa tạo vào Frame :
   fr.add (mPanel);
5. lặp lại các bước 3, 4, 5 để tạo các Panel khác theo yêu cầu và add lần lượt chúng vào Frame.
6. hiển thị Frame và các đối tượng giao diện bên trong nó để người dùng thấy và tương tác được với frame :
   fr.setVisible(true);

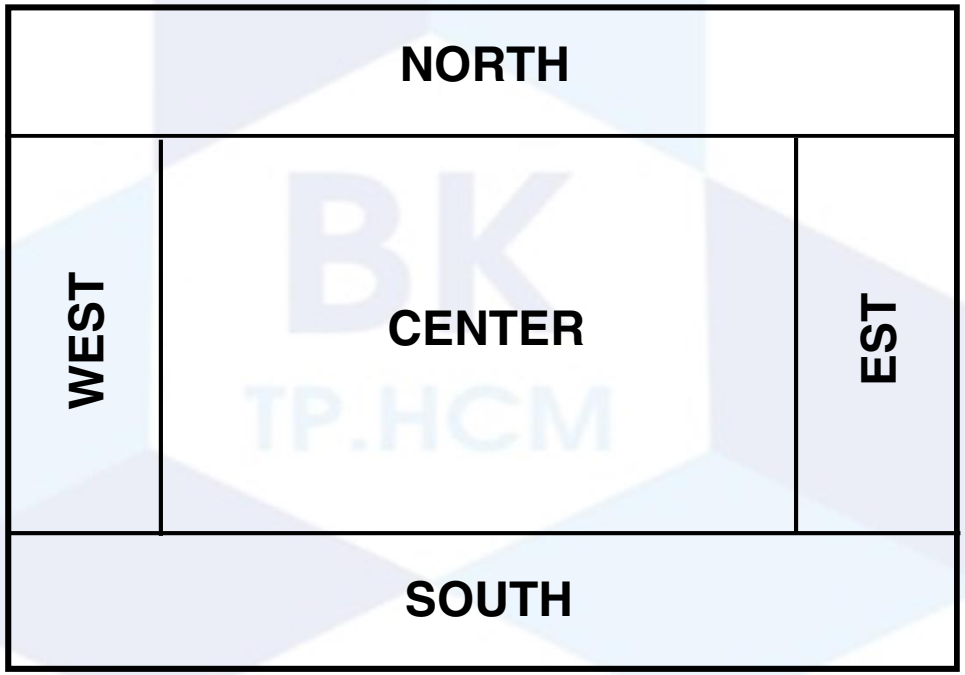
- Thường hai bước 1 và 6 được lập trình trong đoạn code của chương trình, nơi cần tạo và hiển thị Frame. Còn các bước 3, 4, 5 được lập trình trong hàm constructor của Frame tương ứng.

5.4 Cơ chế bố trí các phần tử trong container

- Tác vụ add() để add 1 component vào container không có các tham số xác định độ lớn và vị trí cụ thể của component. Vậy cơ chế nào để máy bố trí các component trong Container ?
- Container không quản lý việc bố trí cụ thể các component trong nó mà nó nhờ 1 đối tượng chuyên trách làm việc này, ta gọi là LayoutManager. Tại từng thời điểm, mỗi Container sẽ kết hợp với 1 LayoutManager và LayoutManager này sẽ quyết định việc bố trí các component bên trong theo cách thức xác định mà nó qui định.
- AWT cung cấp 5 loại LayoutManager khác nhau :
  1. FlowLayout : bố trí các component theo thứ tự chúng được add từ trái sang phải rồi từ trên xuống.

5.4 Cơ chế bố trí các phần tử trong container

2. CardLayout : hiệu chỉnh kích thước và bố trí 2 tới n thành phần (thường là Panel) nằm cùng 1 chỗ, nhưng từng thời điểm chỉ có 1 thành phần hiển thị lên.
3. BorderLayout : hiệu chỉnh kích thước và bố trí các component theo 5 vùng gồm 4 biên + 1 vùng trung tâm của Container :



5.4 Cơ chế bố trí các phần tử trong container

4. GridLayout : thay đổi kích thước và bố trí các component theo dạng bảng nhiều hàng nhiều cột, các phần tử trong bảng đều có cùng kích thước :

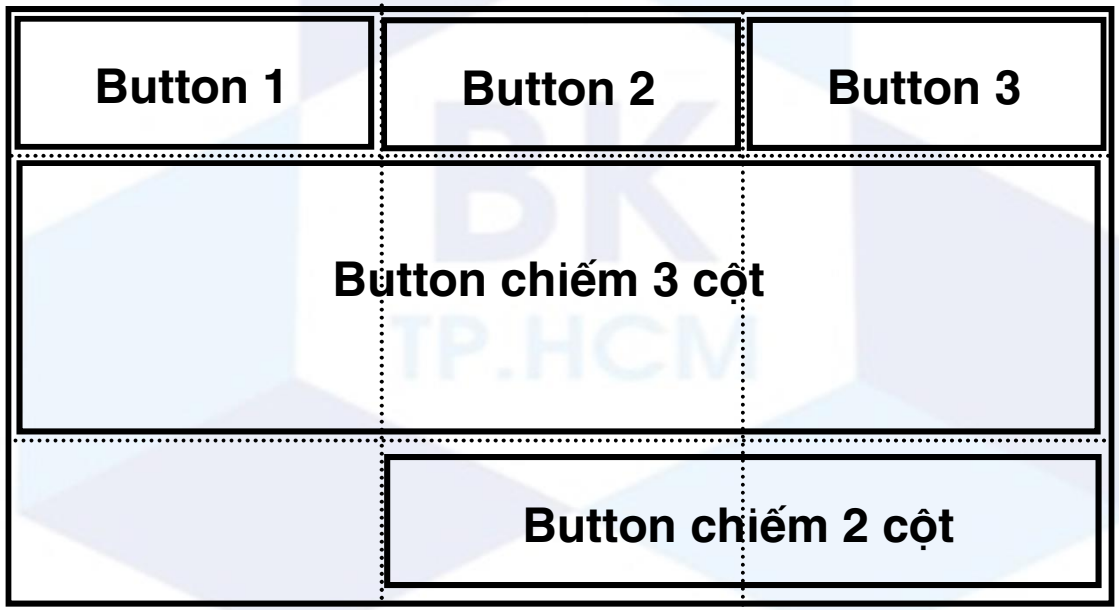
```
//Panel chứa bảng thành phần gồm 3 hàng, 2 cột)
mPanel.setLayout (new GridLayout(3,2));
```

phần tử 1,1	**phần tử 1,2**
phần tử 2,1	**phần tử 2,2**
phần tử 3,1	**phần tử 3,2**

5.4 Cơ chế bố trí các phần tử trong container

5. GridBagLayout : thay đổi kích thước và bố trí các component theo dạng bảng nhiều hàng nhiều cột, nhưng độ cao từng hàng có thể khác nhau, tương tự độ rộng từng cột có thể khác nhau.

   Thí dụ Panel dưới đây có 3 hàng, 3 cột. Hàng đầu chứa 3 button. Hàng 2 chứa 1 button có độ cao gấp đôi và độ rộng chiếm cả 3 cột. Hàng 3 chứa Button chiếm 2 cột bên phải.



5.5 Tương tác giữa người dùng và đối tượng giao diện

- Mỗi phần tử giao diện (Container, Component) không chỉ hiển thị bộ mặt (đồ họa) của mình cho người dùng xem, chức năng thiết yếu khác của nó là giúp người dùng tương tác với chương trình.
- Ta gọi sự kiện là 1 thao tác nào đó mà người dùng thực hiện trên phần tử giao diện cụ thể, thí dụ sự kiện ấn chuột trái xuống, thả chuột trái ra, ấn chuột phải xuống, thả chuột phải ra, dời chuột, ấn 1 phím xuống, thả phím ra,...
- Để xử lý 1 sự kiện trên 1 phần tử giao diện, ta sẽ viết 1 tác vụ chức năng (hàm xử lý sự kiện), thường tác vụ này được viết trong class miêu tả Frame hay Panel chứa đối tượng mà người dùng muốn tương tác. Ta gom nhiều tác vụ xử lý sự kiện có mối quan hệ lẫn nhau thành 1 class, thí dụ class KeyAdapter hiện thực interface KeyListener chứa các tác vụ xử lý sự kiện về phím, class MouseAdapter hiện thực interface MouseListener chứa các tác vụ xử lý sự kiện về chuột.

5.5 Tương tác giữa người dùng và đối tượng giao diện

- Thí dụ sau miêu tả việc định nghĩa 1 vài hàm xử lý sự kiện trên cửa sổ và trên component bên trong nó :

```
public class MyFrame extends Frame {
public MyFrame() { //constructor của class MyFrame
    super("Title of my frame");
    setSize(350, 230);
    //định nghĩa hàm xử lý sự kiện đóng form
    addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent e) {
        System.exit(0);
      }
    });
```

5.5 Tương tác giữa người dùng và đối tượng giao diện

```
//định nghĩa hàm xử lý sự kiện actionPerformed (click chuột)
ActionListener buttonListener = new ActionListener() {
   public void actionPerformed(ActionEvent ae) {
      giaiPTB2(); //tác vụ thực hiện chức năng mong muốn
   }};
//tạo Panel chứa Button và các Component khác
mPanel = new Panel();
mPanel.setLayout(new GridLayout(0, 2));
btnStart = new Button("Bat dau giai");
btnStart.addActionListener(buttonListener);
mPanel.add(btnStart);    //add Button vào Panel
add(mPanel);             //tạo Panel vào Frame
} //hết hàm constructor
```

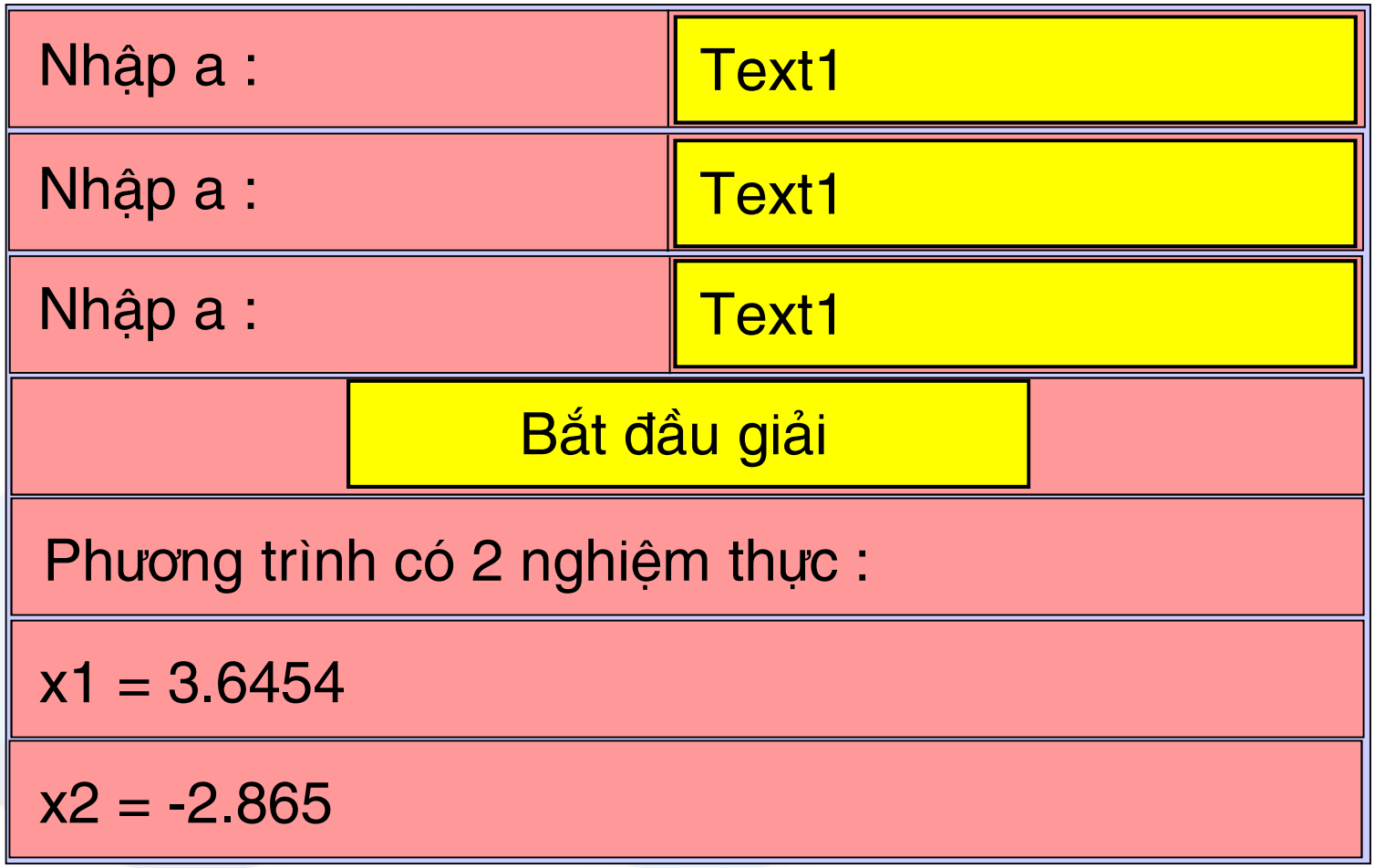
5.6 Thí dụ viết phần mềm giải phương trình bậc 2

- Phân tích chương trình giải phương trình bậc 2 ta thấy cần 1 số đối tượng giao diện sau đây :
  - 1 Label và 1 Textbox để yêu cầu người dùng nhập tham số a.
  - 1 Label và 1 Textbox để yêu cầu người dùng nhập tham số b.
  - 1 Label và 1 Textbox để yêu cầu người dùng nhập tham số c.
  - 1 Button để người dùng ra lệnh giải phương trình.
  - 3 Label để hiển thị kết quả (1 miêu tả kết quả tổng quát và 2 nghiệm).

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

- Ta có thể lập trình để bố trí các đối tượng giao diện như sau :

Frame chứa 7 Panel, 3 Panel trên chứa cặp Label,TextBox, Panel 4 chứa Button, 3 Panel còn lại chứa Label.



5.6 Thí dụ viết phần mềm giải phương trình bậc 2

- Mã nguồn của class đặc tả cửa sổ chương trình giải phương trình bậc 2 :

```
package awtgptb2;
import java.awt.*;
import java.awt.event.*;
public class FrGPTB2 extends Frame {
    //định nghĩa các biến tham khảo đến các đối tượng giao diện
    private Label labelA;
    private TextField txtA;
    private Label labelB;
    private TextField txtB;
    private Label labelC;
    private TextField txtC;
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
private Button btnStart;
private Label lblKetqua;
private Label lblX1;
private Label lblX2;
//xóa nội dung các Textbox và Label kết quả về null
public void init() {
   txtA.setText(null);
   txtB.setText(null);
   txtC.setText(null);
   lblKetqua.setText(null);
   lblX1.setText(null);
   lblX2.setText(null);
}
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//hàm giải phương trình bậc 2 theo 3 tham số nhập
void giaiPTB2() {
 //định nghĩa các biến cần dùng
 double a, b, c;
 double delta;
 double x1, x2;
 //mã hóa dữ liệu chuỗi thành giá trị số Double
 a = Double.valueOf(txtA.getText());
 b = Double.valueOf(txtB.getText());
 c = Double.valueOf(txtC.getText());
 //tính biệt số delta
 delta = b * b - 4 * a * c;
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//kiểm tra biệt số delata và quyết định xử lý
if (delta >= 0) { //trường hợp có 2 nghiệm thực
   x1 = (-b + Math.sqrt(delta)) / 2 / a;
   x2 = (-b - Math.sqrt(delta)) / 2 / a;
   lblKetqua.setText("Phương trình có 2 nghiệm :");
   lblX1.setText("x1 = " + x1);
   lblX2.setText("x2 = " + x2);
} else { //trường hợp vô nghiệm
   lblKetqua.setText("Phương trình vô nghiệm !");
   lblX1.setText("");
   lblX2.setText("");
}
}
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
public FrGPTB2() {  //constructor của frame
   super("Giải phương trình bậc 2");
   setSize(350, 230); //thiết lập kích thước frame
   //định nghĩa hàm xử lý sự kiện đóng Form
   addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent e) {
         System.exit(0);
      }
   });
   //thiết lập LayoutManager cho Frame
   this.setLayout(new GridLayout(7,1));
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//tạo Panel chứa label và textbox A
Panel mPanel = new Panel();
//khai báo layout cho Panel có 2 cột
mPanel.setLayout(new GridLayout(1, 2));
//tạo Label và add nó vào Panel
labelA = new Label("Nhập a : ");
mPanel.add(labelA);
//tạo Textbox và add nó vào Panel
txtA = new TextField("");
mPanel.add(txtA);
//add Panel vào Frame
add(mPanel);
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//tạo Panel chứa label và textbox B
mPanel = new Panel();
//khai báo layout cho Panel có 2 cột
mPanel.setLayout(new GridLayout(1, 2));
//tạo Label và add nó vào Panel
labelB = new Label("Nhập b : ");
mPanel.add(labelB);
//tạo Textbox và add nó vào Panel
txtB = new TextField("");
mPanel.add(txtB);
//add Panel vào Frame
add(mPanel);
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//tạo Panel chứa label và textbox C
mPanel = new Panel();
//khai báo layout cho Panel có 2 cột
mPanel.setLayout(new GridLayout(1, 2));
//tạo Label và add nó vào Panel
labelC = new Label("Nhập c : ");
mPanel.add(labelC);
//tạo Textbox và add nó vào Panel
txtC = new TextField("");
mPanel.add(txtC);
//add Panel vào Frame
add(mPanel);
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//tạo Panel có 3 cột để chứa Button ở giữa
mPanel = new Panel(); mPanel.setLayout(new GridLayout(1, 3));
mPanel.add(new Label(""));
//tạo Button và add nó vào Panel
btnStart = new Button("Bắt đầu giải");
//định nghĩa hàm xử lý click chuột trên Button
btnStart.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent ae) {
      giaiPTB2(); //gọi hàm giải ptb2
   }
});
mPanel.add(btnStart);          //add Button vào Panel
mPanel.add(new Label(""));
add(mPanel);                   //add Panel vào Frame
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
//tạo Panel chứa Label kết quả
mPanel = new Panel();
//khai báo layout cho Panel có 2 cột
mPanel.setLayout(new GridLayout(1, 1));
//tạo Label kết quả và add nó vào Panel
lblKetqua = new Label("");
mPanel.add(lblKetqua);
add(mPanel);      //add Panel vào Frame
//tạo Panel chứa Label x1 =
mPanel = new Panel();
//khai báo layout cho Panel có 2 cột
mPanel.setLayout(new GridLayout(1, 1));
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

```
        //tạo Label x1= và add nó vào Panel
        lblX1 = new Label("");
        mPanel.add(lblX1);
        add(mPanel);      //add Panel vào Frame
        //tạo Panel chứa Label x2 =
        mPanel = new Panel();
        mPanel.setLayout(new GridLayout(1, 1));
        //tạo Label x2= và add nó vào Panel
        lblX2 = new Label("9");
        mPanel.add(lblX2);
        add(mPanel);      //add Panel vào Frame
    }
}
```

5.6 Thí dụ viết phần mềm giải phương trình bậc 2

- Mã nguồn của class chương trình dùng cửa sổ giải phương trình bậc 2 :

```
package awtgptb2;
public class AWTGPTB2 {
    public static void main(String[] args) {
        FrGPTB2 mainFrame = new FrGPTB2();
        //khởi động nội dung ban đầu cho các đối tượng
        mainFrame.init();
        //hiển thị frame để người dùng sử dụng
        mainFrame.setVisible(true);
    }
}
```

5.7 Kết chương

- Chương này đã trình bày các đối tượng giao diện đồ họa được cung cấp trong thư viện AWT (Abstract Windowing Toolkit), tính chất của từng đối tượng, mối quan hệ giữa chúng.
- Chương này cũng đã giới thiệu qui trình lập trình tạo đối tượng Frame miêu tả cửa sổ giao diện cùng các đối tượng giao diện AWT được chứa bên trong nó theo yêu cầu của chương trình. Cuối chương đã trình bày mã nguồn của chương trình giải phương trình bậc 2 dùng các đối tượng giao diện AWT.