- イベント処理

★10　イベント処理 (p.161)

★10.1　イベント処理のしくみ

ユーザが GUI コンポーネントに対して操作（ボタンを押した，メニューの項目を選んだ，キー入力した，マウスカーソルをある場所に移動して，etc.）を行なった（これを「**イベント**が発生した」という）ときに，それに反応するプログラムを作るには，**イベント処理**を組み込む必要がある．

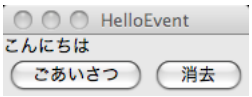
ある所（イベント発生源となるオブジェクト，**イベントソース**）でイベントが発生すると，そのイベントの情報（どんな操作がどの位置で行なわれたかなど）を格納したイベントクラスのオブジェクト（**イベントオブジェクト**）が生成される．このオブジェクトは，イベントソースからそれを処理するオブジェクト（**イベントリスナー**）へと送られる．

イベント処理のプログラムでは，次のようなことをする必要がある．

1. リスナーとなるクラスを定義し，イベントを処理するメソッドを書く
2. ソースとなるオブジェクトにこのリスナーのオブジェクトを登録する

★10.2　はじめてのイベント処理—ボタンのイベント

別の頁に示した HelloEvent.java は，次の図に示すようなイベント処理を実現したものである．



この例では,「ボタンが押された」というイベントの処理を行なっている．p.162 の表を見ると，次のことがわかる．

- 発生するイベントオブジェクトは ActionEvent クラスのものである
- それに対応するリスナーのクラスは ActionListener である

これにしたがって，HelloEvent.java では次のような記述を行なっている．

1. 4 行目で HelloEvent クラスを宣言する際に次のように書いている．
   `public class HelloEvent extends JPanel implements ActionListener`
   これは,「HelloEvent クラスは JPanel のサブクラスであり，ActionListener インタフェース (☆ 1) を実装（インプリメント）している」との宣言である．
2. HelloEvent クラスのインスタンスメソッドとして，`actionPerformed` を定義している（18 から 25 行目）．`actionPerformed` は，ActionListener で定められたイベント処理メソッドである．
3. イベントソースとなるボタンオブジェクトの `addActionListener` メソッドを呼び,「これ」(HelloEvent のオブジェクト）をリスナーとして登録している（11,12 行目）．

☆ 1) インタフェースについてはこの授業ではスキップした．詳細は，教科書 p.88 参照．

このように，イベント処理を行なう場合には，扱いたいイベントの種類に応じた適切なリスナー（p.162 表 13.1 で一覧できる）を選択し，そのリスナーのイ

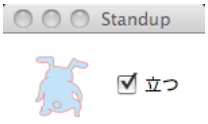
ベント処理メソッド（p.164 表 13.2）を定義する．複数のリスナーを設定して種類の異なる複数のイベント処理を行なうことも可能である．

HelloEvent.java

```
 1  import java.awt.*;
 2  import java.awt.event.*;  // ActionListener のため
 3  import javax.swing.*;

 4  public class HelloEvent extends JPanel implements ActionListener{


 5     JLabel label;    //ラベル用の変数
 6     JButton b1, b2;  //ボタン用の変数

 7     public HelloEvent(){
 8        label = new JLabel(" "); //最初は何も表示しない（""ではなく" "）
 9        b1 = new JButton("ごあいさつ");
10        b2 = new JButton("消去");
11        b1.addActionListener(this);  //イベントリスナーを設定


12        b2.addActionListener(this);  //イベントリスナーを設定


13        setLayout(new BorderLayout());
14        add(label, BorderLayout.NORTH);
15        add(b1, BorderLayout.CENTER);
16        add(b2, BorderLayout.EAST);
17     }

18    public void actionPerformed(ActionEvent e){



19       Object obj = e.getSource();




20       if(obj == b1){                // b1 が押されたとき
21          label.setText("こんにちは"); // label に文字列を設定
22       }
23       else if(obj == b2){           // b2 が押されたとき
24          label.setText(" ");        // label の文字列を消す
25       }
26     }

       (main メソッドはおなじみの形なので省略)
35  }
```

★10.3 おかわり—チェックボックスのイベント

以下は，前頁のものとは異なるイベント処理の例である．



Standup.java（p.168 のものとは少し異なる）

```
（import 文は省略）
4
5  public class Standup extends JPanel implements ItemListener{

6      JLabel label;      //ラベル
7      JCheckBox cb;      //チェックボックス
8      ImageIcon lay,up; //アイコン

9      public Standup(){
10        setBackground(Color.white); //パネルの背景を白に
11        lay = new ImageIcon("lay.gif");
12        up = new ImageIcon("up.gif");
13        label = new JLabel(lay);
14        cb = new JCheckBox("立つ");
15        cb.setBackground(Color.white);//チェックボックスの背景を白に
16        cb.addItemListener(this);      //リスナーの設定
17        add(label);                    //ラベルを置く
18        add(cb);                       //チェックボックスを置く
19     }
20

21     public void itemStateChanged(ItemEvent e){
22         if(cb.isSelected()){
23             label.setIcon(up);  // 選択された状態なら up を指定
24         }else{
25             label.setIcon(lay);  // さもなくば lay を指定
26         }
27     }
        （main メソッドは省略）
37  }
```

上記のイベント処理メソッドは，次のようにも書ける．ただし，こちらの場合，どのチェックボックスでイベントが発生したかを調べていないことに注意．

別バージョンの itemStateChanged

```
21     public void itemStateChanged(ItemEvent e){
22        if( e.getStateChange()==ItemEvent.SELECTED){ //選択された状態

23           label.setIcon(up);
24        }else if(e.getStateChange()==ItemEvent.DESELECTED){ //非選択の状態
25           label.setIcon(lay);
26        }
27     }
```

例は示さないが，複数のチェックボックスで発生するイベントを処理をする場合は，イベントオブジェクトの getSource メソッドと上記のいずれかの処理を組み合わせることになるだろう．

Q1. 以下のプログラム (☆ 2) を実行すると，ボタンとラベルが上下に並んだウィンドウが表示される．これを改造して,「最初はラベルに「0 回」と表示されてる」→「ボタンを押すたびに「1 回」「2 回」と数字が増えていく」，というものにしよう．

☆ 2) このプログラムは先の例と違い，this をきちんと付けた書き方をしている．

Counter.java

```
 1  import java.awt.*;
 2
 3  import javax.swing.*;
 4
 5  public class Counter extends JPanel{
 6      JButton button;
 7      JLabel  label;
 8
 9      public Counter(){
10          this.button = new JButton("押してちょ");
11          this.label  = new JLabel("ほげ");
12
13          this.setLayout(new BorderLayout());
14          this.add(button, BorderLayout.NORTH);
15          this.add(label,  BorderLayout.CENTER);
16      }
17
18
19
20
21
22
23
        (main メソッドは省略)
32  }
33
```

宿題？

- 授業中に解説されなかった Q を解いてみよう．
- 次回は 13.3 節（13 章のそれ以降の内容はスキップ）です．あらかじめ読んでおくこと．例題のプログラムを作成し実行してみておくこと．