# Java/Swingについて (2)

2005年10月11日

海谷 治彦

# 目次

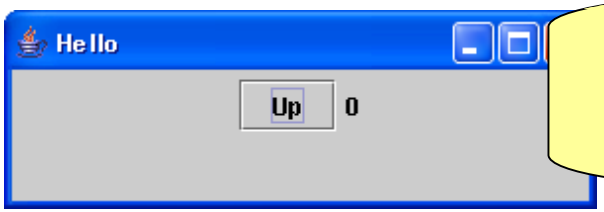
- Adapterについて
- TextField
- TextArea
  - Copy&Paste
- JList
- JComboBox
- JScrollPane
- レイアウトについて

前回より抜粋

# ソースコード

```
public class Listener1 {
  public static void main(String[] args){
    JFrame jf=new JFrame("Hello");
    jf.setSize(300, 100);
    JPanel panel=new JPanel();
    jf.setContentPane(panel);

    JButton button=new JButton("Up");
    panel.add(button);
    CounterLabel counter=new CounterLabel();
    panel.add(counter);

    button.addMouseListener(counter);

    jf.setVisible(true);
  }
}
```

```
public class CounterLabel extends JLabel
                          implements MouseListener {
    private int c=0;

    CounterLabel(){ super(0+""); }
    public void mouseClicked(MouseEvent arg0) {}
    public void mouseEntered(MouseEvent arg0) {}
    public void mouseExited(MouseEvent arg0) {}
    public void mouseReleased(MouseEvent arg0) {}

    public void mousePr    d(MouseEvent arg0) {
      c++;
      setText(c+"");
    }
}
```




使ってないメソッドも記述しないといけないのは無駄，とはいえJavaの文法上省けない．

# Adapter

- Listenerを空実装してあるクラス
- コイツのサブクラスを作成すれば必要なメソッドのみオーバーライドすればよい.
- 一般にListenerを使うよりプログラムが短くなる.

- 詳細はマニュアルページを見て

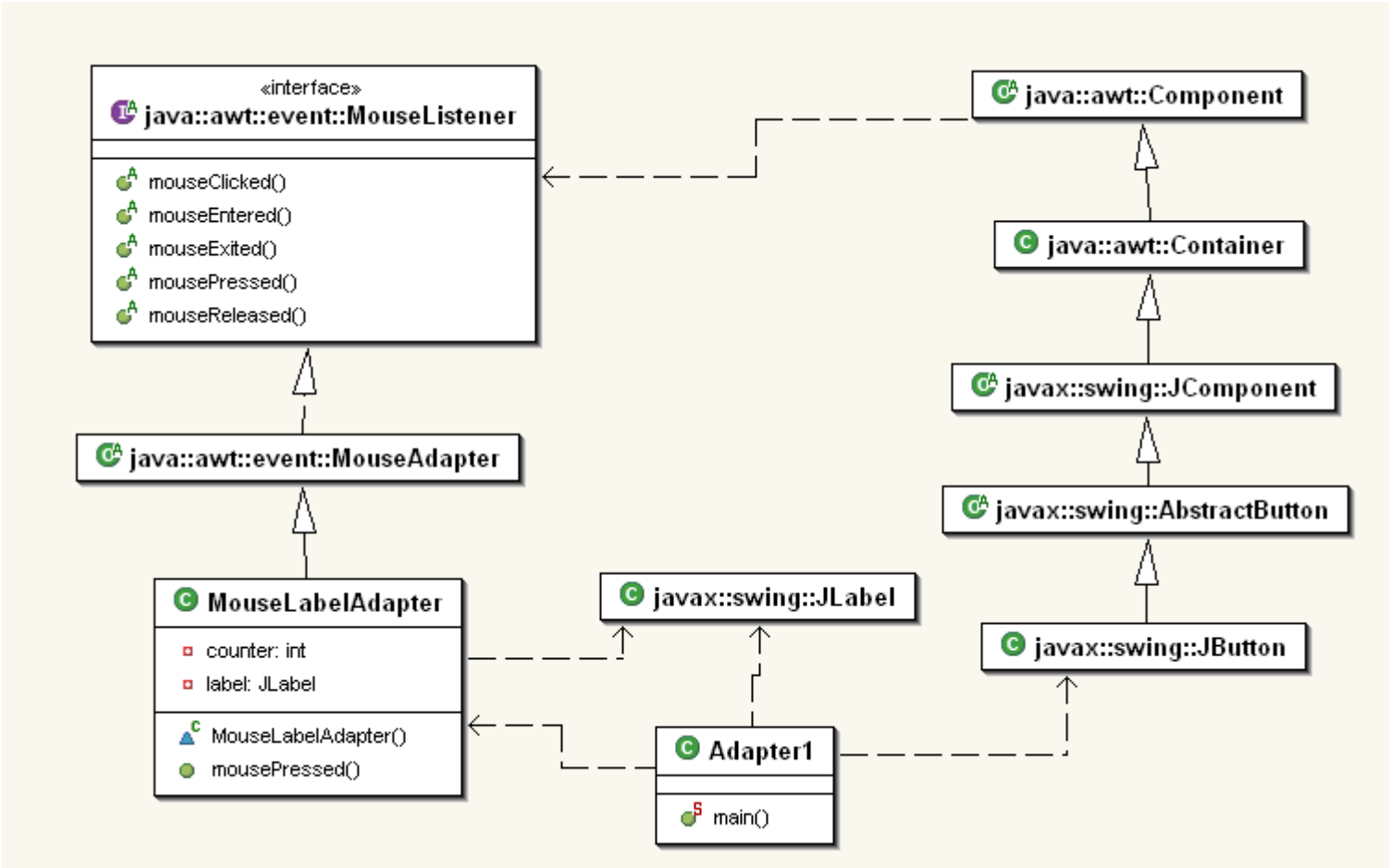
# ソース

```
public class Adapter1 {

  public static void main(String[] args) {
    JFrame frame=new JFrame();
    frame.setSize(300,100);
    JPanel panel=new JPanel();
    frame.setContentPane(panel);

    JButton button=new JButton("Up");
    JLabel label=new JLabel("0");
    panel.add(button);
    panel.add(label);
    MouseLabelAdapter adapter=new MouseLabelAdapter(label);
    button.addMouseListener(adapter);

    frame.setVisible(true);
  }
}
```

```
public class MouseLabelAdapter
                extends MouseAdapter {
  private JLabel label;
  private int counter=0;
  MouseLabelAdapter(JLabel label){
    this.label=label;
  }
  public void mousePressed(MouseEvent e){
    counter++;
    label.setText(counter+"");
  }
}
```

mousePressed以外は実装していない．(スーパークラスで空実装されている)

# クラス図


«interface»
java::awt::event::MouseListener
mouseClicked()
mouseEntered()
mouseExited()
mousePressed()
mouseReleased()
java::awt::Component
java::awt::Container
javax::swing::JComponent
javax::swing::AbstractButton
javax::swing::JButton
java::awt::event::MouseAdapter
MouseLabelAdapter
counter: int
label: JLabel
MouseLabelAdapter()
mousePressed()
javax::swing::JLabel
Adapter1
main()

# JTextField

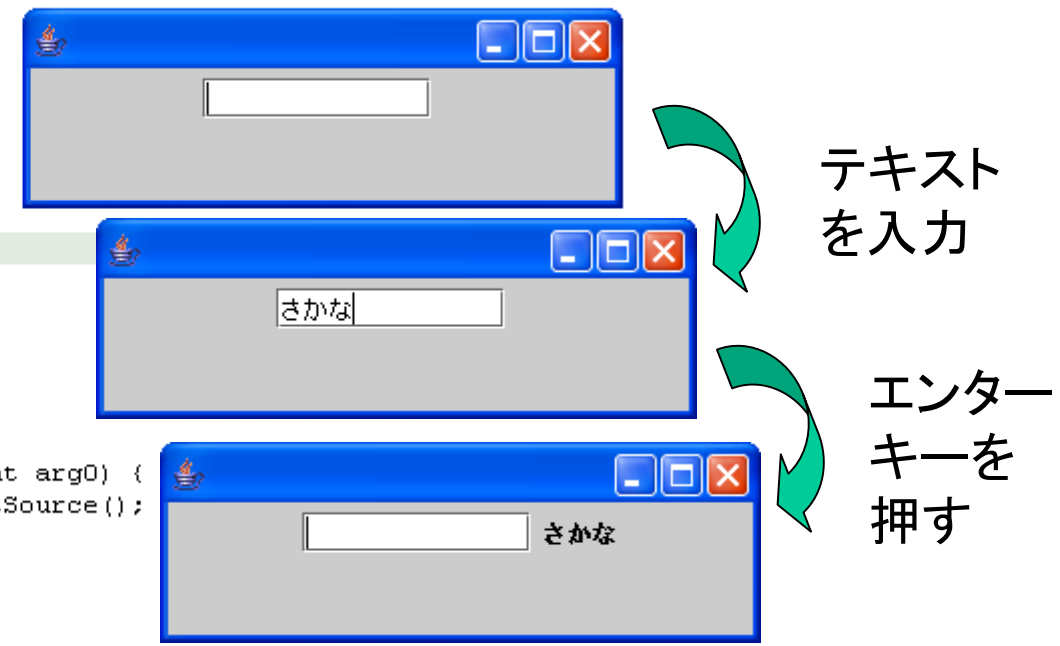
- 一行入力のための部品
- 初期化の方法はいろいろ
  - 文字幅を指定
  - 初期文字列を指定等
- interface ActionListenerによってエンターキー入力を検知できる.

# 例1

```
 */
public class Text1 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(300,100);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        JTextField tf1=new JTextField(10);
        panel.add(tf1);
        JTextField tf2=new JTextField("please fill in.");
        panel.add(tf2);

        frame.setVisible(true);
    }
}
```

# 例2: 入力をラベルに渡す

```
public class Text2 extends JLabel implements ActionListener {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(300,100);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        JTextField tf1=new JTextField(10);
        panel.add(tf1);
        Text2 label=new Text2();
        panel.add(label);

        tf1.addActionListener(label);

        frame.setVisible(true);
    }

    public void actionPerformed(ActionEvent arg0) {
        JTextField tf=(JTextField)arg0.getSource();
        setText(tf.getText());
        tf.setText("");
    }
}
```

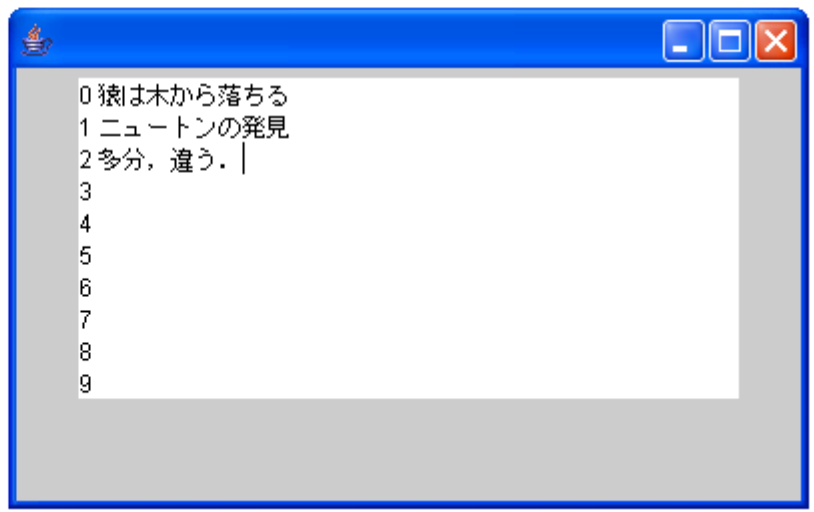



非常に安直にユーザーのテキスト入力を得られる.
(多分，標準入力を使うよりかなり楽)

# JTextArea

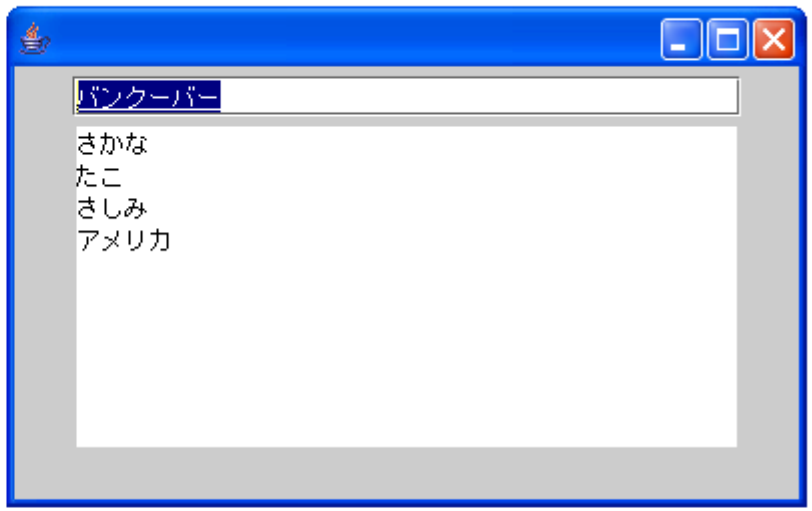
- それ自身，小さなテキストエディタのようなもの．

```
 */
public class Text3 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(400,250);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        JTextArea tf1=new JTextArea(10,30); // line, column
        panel.add(tf1);

        frame.setVisible(true);
    }
}
```



# 例: Field入力をAreaに溜める

```
public class Text3 implements ActionListener{
    private JTextArea area;

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(400,250);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        JTextField field=new JTextField(30);
        panel.add(field);
        JTextArea area=new JTextArea(10,30); //
        panel.add(area);
        Text3 listener=new Text3(area);
        field.addActionListener(listener);

        frame.setVisible(true);
    }

    Text3(JTextArea area){
        this.area=area;
    }
    public void actionPerformed(ActionEvent arg0) {
        JTextField f=(JTextField)arg0.getSource();
        area.append(f.getText()+System.getProperty("line.separator"));
        f.setText("");
    }
}
```

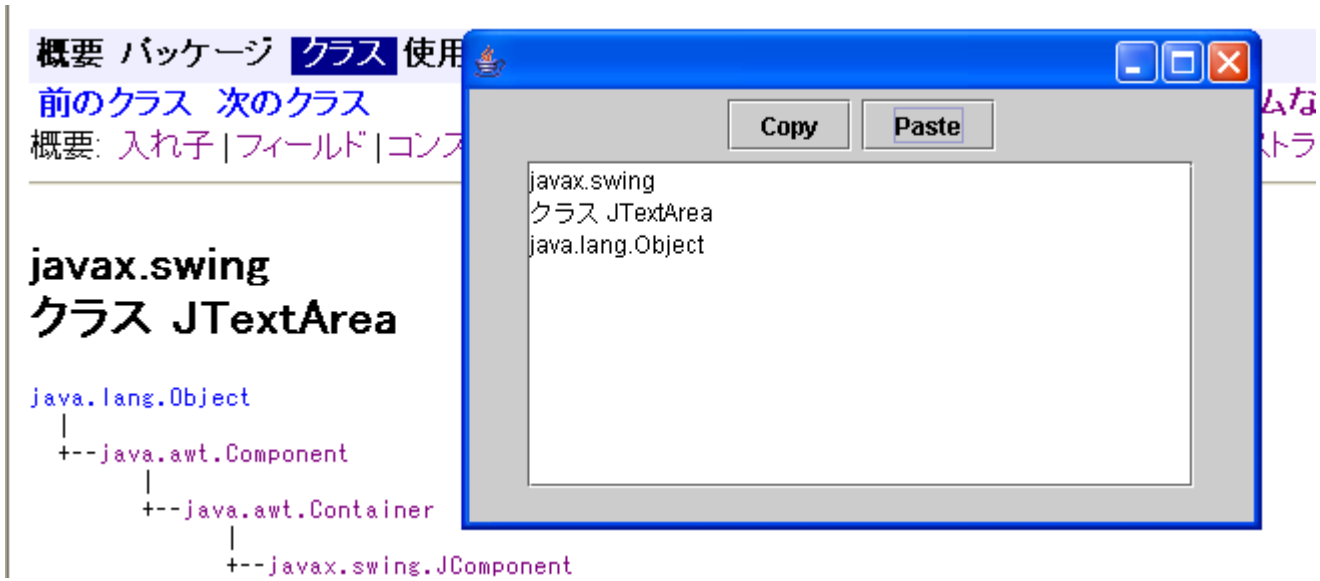


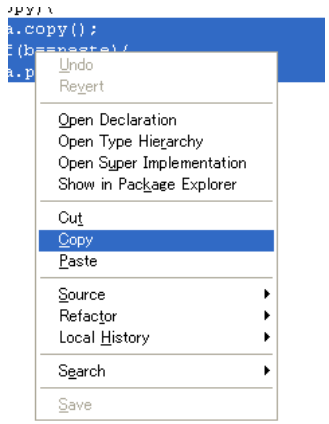
# Copy&Pasteはプログラミングが必要

```
public class ClipAdapter extends MouseAdapter {
  JButton copy; JButton paste; JTextArea area;

  ClipAdapter(JTextArea area, JButton copy, JButton paste){
    this.area=area;
    this.copy=copy;
    this.paste=paste;
  }
  public void mousePressed(MouseEvent e){
    JButton b=(JButton)e.getSource();
    if(b==copy){
      area.copy();
    }else if(b==paste){
      area.paste();
    }
  }
}
```

```
public class Text4 {
  public static void main(String[] args) {
    // 中略
    JButton copy=new JButton("Copy");
    JButton paste=new JButton("Paste");
    JTextArea area=new JTextArea(10, 30);

    ClipAdapter clip=new ClipAdapter(area, copy, paste);
    copy.addMouseListener(clip);
    paste.addMouseListener(clip);

    panel.add(copy);
    panel.add(paste);
    panel.add(new JScrollPane(area));
    frame.setVisible(true);
  }
}
```

# 画面例




かっちょいいメニュー(例: 左図)で
Copy&Pasteできるようにするには,
それなりにコードをかかないといけない.

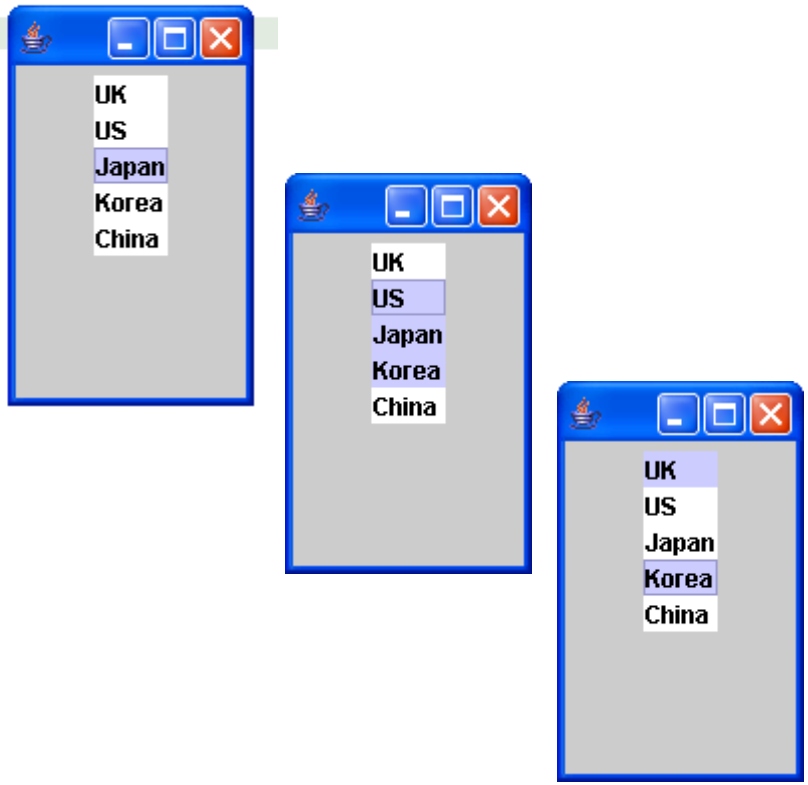
とはいえ, ショートカットキーは有効.

# JList

- 複数の選択肢を列挙するための部品.
- 単一選択，複数選択の双方ができる.
  - 今回は単一選択しか扱わない.
- 無論，選択した項目を取り出せる.
- ListSelectionListener

# 例

```
 */
public class List1 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(100,200);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        String[] listargs={
            "UK", "US", "Japan", "Korea", "China"
        };
        JList list=new JList(listargs);
        panel.add(list);

        frame.setVisible(true);
    }
}
```



# 例

```
 */
public class List2 extends JLabel implements ListSelectionListener {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(100,200);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        String[] listargs={
            "UK", "US", "Japan", "Korea", "China"
        };
        JList list=new JList(listargs);
        // list.setSelectionMode(ListSelectionModel.SINGLE_S
        panel.add(list);
        List2 label=new List2();
        panel.add(label);
        list.addListSelectionListener(label);

        frame.setVisible(true);
    }

    public void valueChanged(ListSelectionEvent arg0) {
        JList list=(JList)arg0.getSource();
        if(arg0.getValueIsAdjusting() == false){ // not multiple
            String s=(String)list.getSelectedValue();
            setText(s);
        }
    }
}
```

# JComboBox

- JListと用途は似ており，項目選択に使える．

```
 *
 * To change the template for this generate
 * Window&gt;Preferences&gt;Java&gt;Code Ge
 */
public class ComboBox1 {

    public static void main(String[] args)
        JFrame frame=new JFrame();
        frame.setSize(300,100);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        String[] comboargs={"UK", "US", "Japan", "Korea", "China"};
        JComboBox combo=new JComboBox(comboargs);
        panel.add(combo);

        frame.setVisible(true);
    }
}
```

# 選択した項目を得られます

```
public class ComboBox2 extends JLabel implements ActionListener{

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(300,100);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);

        String[] comboargs={"UK", "US", "Japan", "Korea", "China"};
        JComboBox combo=new JComboBox(comboargs);
        panel.add(combo);
        ComboBox2 label=new ComboBox2();
        label.setText(combo.getSelectedItem()+"");
        panel.add(label);
        combo.addActionListener(label);

        frame.setVisible(true);
    }

    public void actionPerformed(ActionEvent arg0) {
        JComboBox c=(JComboBox)arg0.getSource();
        setText(c.getSelectedItem()+"");
    }
}
```

# スクロールバー

- JScrollPane()クラスで包むとスクロール可能となる.
- ただし，なんでもスクロールできるわけでない.
  - インタフェース Scrollableを実装した部品.
  - もしくは，「望ましい」サイズが設定されている部品.

    javax.swing.JComponent.getPreferredSize()および setPreferredSize() メソッド参照.

# スクロールできそうな部品

javax.swing

## インタフェース Scrollable

既知の実装クラスの一覧:

JList, JTable, JTextComponent, JTree

javax.swing.text

## クラス JTextComponent

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.text.JTextComponent
```

すべての実装インタフェース:

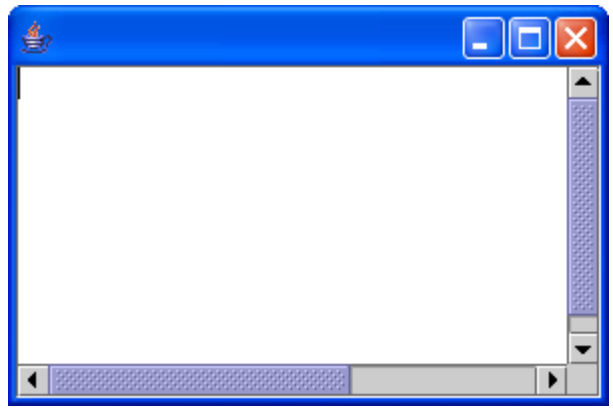
Accessible, ImageObserver, MenuContainer, Scrollable, Serializable

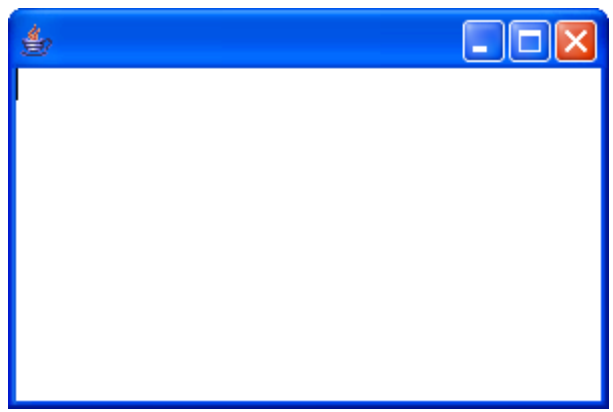
直系の既知のサブクラス:

JEditorPane, JTextArea, JTextField

# 例

```
import javax.swing.*;

public class Scroll2 {
  public static void main(String[] args) {
    JFrame frame=new JFrame();
    frame.setSize(300,200);
    JTextArea area=new JTextArea(10,40);
    JScrollPane scroll=new JScrollPane(area);

    frame.getContentPane().add(scroll);

    frame.setVisible(true);
  }
}
```





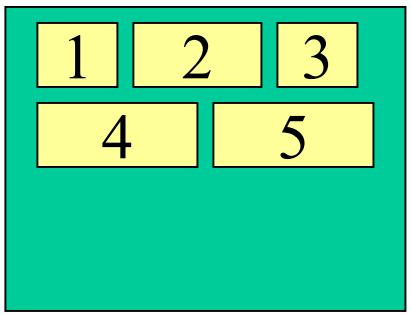
包まないと下のようになる

# 部品のレイアウト制御

- add()で追加された部品が，左から右に追加されるだけでは芸がない．
- LayoutManagerを実装したクラスを使って，いくつかのレイアウトを指定することができる．
- 残念ながらSwing内のレイアウトの種類は多彩とは言い難い．

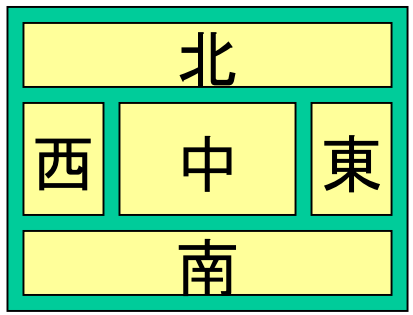
# レイアウトの例

FlowLayout



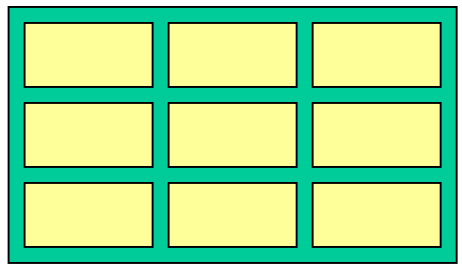

JPanelではデフォルト

BorderLayout



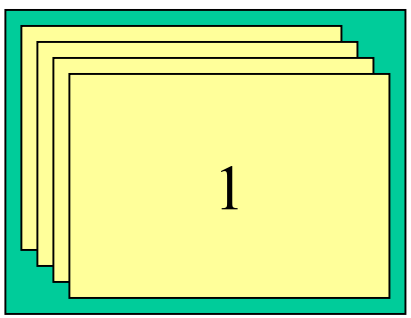

東西南北中の範囲で位置指定可能
JFrameのデフォルトPaneはこのレイアウト

GridLayout



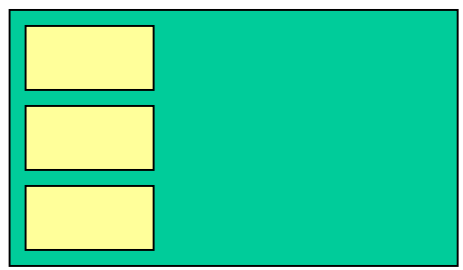
碁盤の目のようにつめる

CardLayout




スライドやトランプのように重ねて表示.

BoxLayout



縦もしくは横一列に並べる.

# レイアウト設定の手順

- なんとかLayoutのインスタンスを作る.
- それを，setLayout()メソッドに与える.
  - 本メソッドはContainerクラスで定義されてる.

**javax.swing**
**クラス JPanel**

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
```

クラス java.awt.Container から継承したメソッド

add, add, add, add, add, addContainerListener, addI
areFocusTraversalKeysSet, countComponents, deliverE
findComponentAt, getComponent, getComponentAt, getC
getComponents, getContainerListeners, getFocusTrave
getLayout, insets, invalidate, isAncestorOf, isFocu
isFocusTraversalPolicySet, layout, list, list, loca
preferredSize, printComponents, processContainerEve
removeAll, removeContainerListener, setFocusCycleRo
setFocusTraversalPolicy, setLayout, transferFocusBa
validate, validateTree

すべての実装インタフェース:
Accessible, ImageObserver, MenuContainer, Serializable

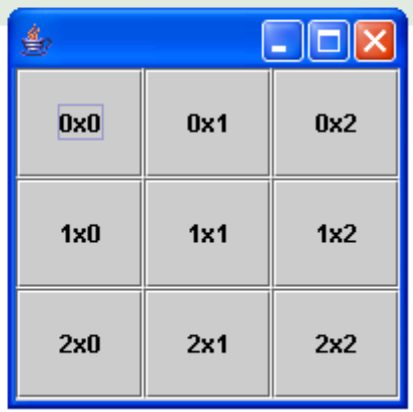
直系の既知のサブクラス:
AbstractColorChooserPanel, JSpinner.DefaultEditor

# 例 BorderLayout

```
/**
 * @author kaiya
 *
 * To change the template for this generated typ
 * Window&gt;Preferences&gt;Java&gt;Code Generat
 */
public class Layout1 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(500,300);
        JPanel panel=new JPanel();
        frame.setContentPane(panel);
        panel.setLayout(new BorderLayout());

        JTextField tf1=new JTextField(10);
        panel.add(tf1, BorderLayout.NORTH);
        JTextArea area=new JTextArea(10,10);
        panel.add(new JScrollPane(area), BorderLayout.CENTER);
        panel.add(new Label("hello"), BorderLayout.SOUTH);

        String[] ss={"US", "UK", "Japan", "Korea", "China"};
        JList list=new JList(ss);
        panel.add(list, BorderLayout.WEST);
        JComboBox combo=new JComboBox(ss);
        panel.add(combo, BorderLayout.EAST);

        frame.setVisible(true);
    }
}
```

# 例 GridLayout

```
 */
public class Layout3 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(200, 200);
        JPanel panel=(JPanel)frame.getContentPane();
        panel.setLayout(new GridLayout(3,3));

        for(int i=0; i<3; i++)
            for(int j=0; j<3; j++){
                panel.add(new JButton(i+"x"+j));
            }

        frame.setVisible(true);
    }
}
```

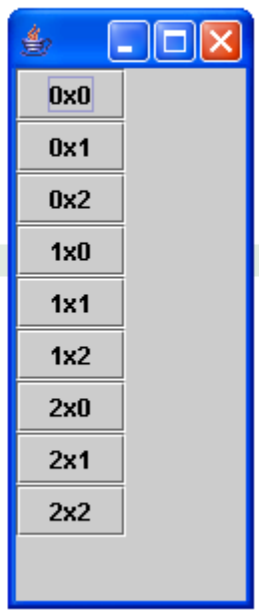



まぁ，ありがちの例だが・・・

# 例 BoxLayout

swingパッケージに含まれることに注意

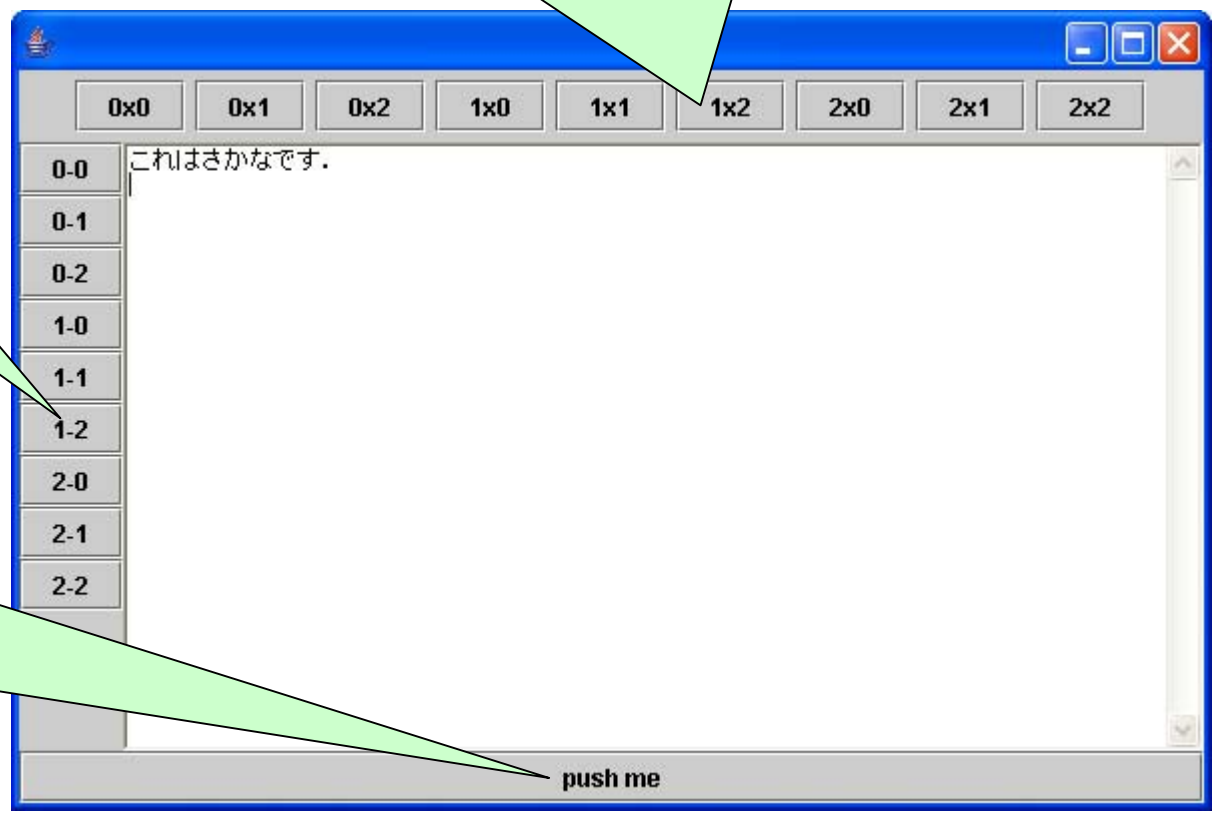
```
import javax.swing.*;

public class Layout4 {

    public static void main(String[] args) {
        JFrame frame=new JFrame();
        frame.setSize(100, 300);
        JPanel panel=(JPanel)frame.getContentPane();
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

        for(int i=0; i<3; i++)
            for(int j=0; j<3; j++){
                panel.add(new JButton(i+"x"+j));
            }

        frame.setVisible(true);
    }
}
```



# 例 Layoutの入れ子

WESTにはBoxLayoutでボタンを縦に9個単純に並べた.

NORTHにはFlowLayoutでボタンを9個単純に並べた.




直接ボタンをSOUTHにのせると一杯に広がる

全体はBorderLayout (デフォルトのまま)

# ソース

```
public class Layout5 {
  public static void main(String[] args) {
    JFrame frame=new JFrame();
    frame.setSize(600, 400);
    JPanel panel=(JPanel)frame.getContentPane(); // レイアウトはデフォのBorderを利用
    JPanel npanel=new JPanel(); // レイアウトはデフォルトのFlowを利用
    JPanel wpanel=new JPanel();
    wpanel.setLayout(new BoxLayout(wpanel, BoxLayout.Y_AXIS));

    for(int i=0; i<3; i++) for(int j=0; j<3; j++){
        npanel.add(new JButton(i+"x"+j));
        wpanel.add(new JButton(i+"-"+j));
    }
    panel.add(npanel, BorderLayout.NORTH);
    panel.add(wpanel, BorderLayout.WEST);
    panel.add(new TextArea(20,40), BorderLayout.CENTER);
    panel.add(new JButton("push me"), BorderLayout.SOUTH);
    frame.setVisible(true);
  }
}
```

# レイアウトの無効化

- 既存のレイアウトは，任意の場所にボタン等を配置することを許さない.
- ボタン等を任意配置したい場合は，レイアウト自体を無効化する必要がある.
- 無効化すると，配置場所だけでなく，サイズも含めて，手動で設定しないといけない (ので少し面倒).
- 例題: web page参照
  - 単にレイアウトを無効化
  - マウスに追従する部品
  - 上記とScrollPaneの連携

# まとめ

- CGIページのような部品は一通り紹介した.
- これらの部品間にイベント送付の関係をつければ，そこそこ快適な入出力機能(ユーザーインタフェース)が作成可能.