

9 (Swing)

- Model-View-Controller
-
-
-
-
-
-
-
-
-

GUI(Graphic User Interface)

GUI

가

(swing)

가

(architecture)

look and feel

AWT

가

2

(frame work)

Kim Topley Core Java Foundation Classes (Prentice Hall 1998)

- -

(Model-View-Controller Design Pattern)

(Swing Component

Architecture)

(Tree)

가

가

(

)

(content)

(visual appearance)

(behavior) :

가

look and feel

. Metal

Window

Motif

가

가

가

look and feel

5

가

look and feel

(content)

Controller: MVC)

가

(Model-View-

3가

(Model): (content)

(View):

(Controller): (Input)

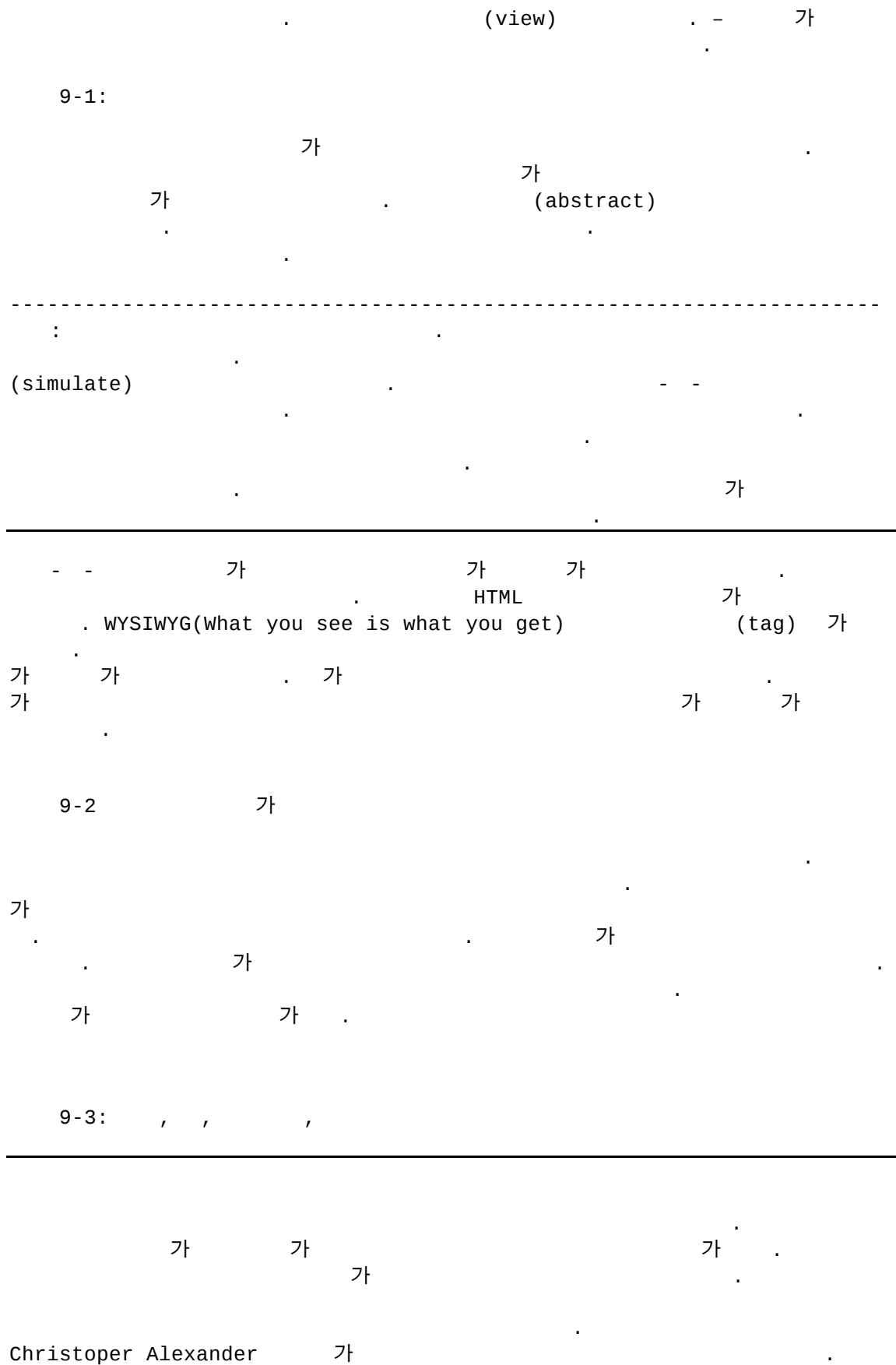
3가

(content)

(active)

(tag)

가



Timeless way of Building(Oxford University Press)
Alexander

(catalog)

가 가
가
1.
2.
9-4
가 가
가
가
가
가
look and feel
3 가
가
Elements of
Reusable Object-Oriented Software by Erich Gamma
- A system of Patterns by Frank Buschmann John
Wiley & sons 1996
AWT
listener
가
listener

(wrapper) 가 가 (JButton JTextField
) (wrapper) (caret)
가 가
) 가 가

. (. look and feel
 .)
 - 가 가 (pluggable) look and feel 가
 look and feel
 .
 .
 가
 (pluggable) look and feel
 가 가
 가
 가
 - (wrapper)
 - 가
 가
 model
 가
 ButtonModel 가
 DefaultButtonModel
 ButtonModel
 . Table 9-1 (accessor)

9-1 ButtonModel

GetActionCommand ()	(action) 가
getMnemonic()	
isArmed()	가 true
isEnabled()	가 true
isPressed()	true
isRollover()	가 true
isSelected()	(toggle) true - (radio button) (check box)

JButton

```

JButton button = new JButton("Blue");
ButtonMode modeel = button.getModel ( );
    
```

- JButton 가 . (

ButtonModel

```
(DefaultButtonModel),
가 가
가 . metal look and feel JButton
ButtonUIListener BasicButtonUI
UI 가
JButton JButton
가 JButton
(DefaultButtonModel),
DefaultButtonUI JComponent
```

(layout)

(form) (designer)가 . JDK VB Delphi
Java-enabled
가 look and feel

9-5: 3

가

```
1.
JButton yellowButton = new JButton("Yellow")

2.
add(yellowButton);

3.
yellowButton.addActionListener(this);
```

? 9-6 6

가

9-6: (flow)

6

가 가 가 . (9-7)

9-7

(dynamic) 가 가 .:

(default)


```

        add(yellowButton, "South");
    }

```

```

        가 center 가 (center)
        가 , 가
        가
        "Center" 가 (
        가 가 "Center"
        가
        가
        BorderLayout 가
        JFrame (content pane) 가
        center

```

```

Container contentPane = getContentPane( );
contentPane.add(yellowButton, "South");

```

```

api: java.awt.Container
void add(Component c, Object constraints)

```

```

:
c
constraints (identifier)

```

```

api: java.awt.BorderLayout(int hgap, int vgap)
, 가 BorderLayout

```

```

:
hgap ( (overlap) .)
vgap ( (overlap) .)

```

```

BorderLayout 가 9-9 (southern)

```

```

가 (panel)
가 가
(prototype)

```

```

GridBagLayout

```

9-9

VB : 가 (picture box)

9-10 .
(south) .

9-10 .

9-10 가 .
JPanel (instance) .
FlowLayout .
add .
FlowLayout

. 3 가 가

```
Container contentPane = getContentPane ( );
JPanel panel new JPanel ( );
panel.add(yellowButton);
panel.add(blueButton);
panel.add(redButton);
contentPane.add(panel, "South");
```

: .
(organizee)

JPanel FlowLayout .
setLayout . JPanel ()

api)
javax.swing.JPanel
JPanel (LayoutManager m)

가 .
(edit) . 2 가 가
text fields text areas . text field
text area .
JTextField JTextArea
JTextComponent . JTextComponent
API 가 가
JTextComponent . API 가 가
field area 가

JTextComponent

api: javax.swing.JTextComponent

void setText(String t)

t

String getText()

void setEditable (boolean b)

가 JTextComponent

(text field)

가

가

JPanel panel = new JPanel();

JTextField textField = new JTextField("Default input", 20);

panel.add(textField);

가 "Default input"

20 " " ,

(column)

n

n

1 2

AWT

가

가

JTextField

가

가

(run time)

setColumn 가

:

setColumns

가

validate

가

textField.setColumns(10);

validate();

validate

validate

가

,

가

JTextField

```

        JTextField textfield = new JTextField("", 20);

        TextComponent setText
            .
            ,

        hourField.setText("12");

        getText 가 . 가
            trim . getText
        String hour = hourField.getText().trim();

        , java.awt.Component setFont
            .
            가 . 9-11 2 가
            . (update) .
            9-11 .

            . ( )

        .: (the model)
        (view) . 가 .
            가 (architecture)가
            가 (formated) ,
            ( HTML) Document
        document listener 가 document 가

        textField1.getDocument ( ).addDocumentListener (listener) ;

        가 가 가 가 .

        void insertUpdate(DocumentEvent e)
        void removeUpdate(DocumentEvent e)
        void changedUpdate(DocumentEvent e)

            가 .

            가 . 가

        (callback) .-
        (adapter) . listener 3 가

        class TextTestFrame extends JFrame
        implements DocumentListener
        {
            public void insertUpdate(DocumentEvent e)
            { setClock();

```

```

    }
    public void removeUpdate(DocumentEvent e)
    { setClock();
    }
    public void changedUpdate(DocumentEvent e)
    {
    }
}

setClock      getText
가              (integer)

int hours = Integer.parseInt(hourField.getText().trim());
int minutes = Integer.parseInt(minuteField.getText().trim());

가              "two"

java.lang.NumberFormatException

DocumentEvent      qusghkeehls

가      가 8
ActionEvent DocumentEvent
DocumentEvent      EventObejct      가
가              getSource

public void insertUpdate(DocumentEvent e)
{ Document d = e.getDocument ( );
. . .
}

가?      가 (editablee) -
HTML      가
(structure)      (reference)
(reference)

Document d = e.getDocument( );
int length = d.getLength( );
String text = null;
try
{
    text = d.getText(0, length);
}
catch(BadLocationException ex) { }

가

9-1
(listener)      가
..

-----

```

```

:
(listener) . 가 ENTER
.
. 가 ENTER
.
(focus) .
-----

```

9-1:TextTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

class TextTestFrame extends JFrame
    implements DocumentListener
{
    public TextTestFrame()
    {
        setTitle("TextTest");
        setSize(300, 200);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        Container contentPane = getContentPane();

        JPanel p = new JPanel();
        hourField = new JTextField("12", 3);
        p.add(hourField);
        hourField.getDocument().addDocumentListener(this);

        minuteField = new JTextField("00", 3);
        p.add(minuteField);
        minuteField.getDocument().addDocumentListener(this);

        contentPane.add(p, "South");
        clock = new ClockPanel();
        contentPane.add(clock, "Center");
    }

    public void insertUpdate(DocumentEvent e)
    {
        setClock();
    }
    public void removeUpdate(DocumentEvent e)
    {
        setClock();
    }
    public void changedUpdate(DocumentEvent e)
    {
    }

    public void setClock()
    {
        int hours
            = Integer.parseInt(hourField.getText().trim());
    }
}

```

```

        int minutes
        = Integer.parseInt(minuteField.getText().trim());
        clock.setTime(hours, minutes);
    }

    private JTextField hourField;
    private JTextField minuteField;
    private ClockPanel clock;
}

class ClockPanel extends JPanel
{
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.drawOval(0, 0, 100, 100);
        double hourAngle
            = 2 * Math.PI * (minutes - 3 * 60) / (12 * 60);
        double minuteAngle
            = 2 * Math.PI * (minutes - 15) / 60;
        g.drawLine(50, 50,
            50 + (int)(30 * Math.cos(hourAngle)),
            50 + (int)(30 * Math.sin(hourAngle)));
        g.drawLine(50, 50,
            50 + (int)(45 * Math.cos(minuteAngle)),
            50 + (int)(45 * Math.sin(minuteAngle)));
    }

    public void setTime(int h, int m)
    {
        minutes = h * 60 + m;
        repaint();
    }

    private int minutes = 0;
}

public class TextTest
{
    public static void main(String[] args)
    {
        JFrame frame = new TextTestFrame();
        frame.show();
    }
}

```

API: java.awt.Component

- void validate ()

API: java.awt.JTextField

- JTextField(cols)
 가 JTextField ..
 :
 cols
- JTextField(String text, int cols)
 가 가 JTextField ..

:
text
cols

- void setColumns(int cols)

:
cols

API)

javax.swing.text.Document

- int getLength()

- String getText(int offset, int length)

: offset
 length

- void addDocumentListener(DocumentListener listener)
가

API: javax.swing.event.DocumentEvent

- Document getDocument()

API)

javax.swing.event.DocumentListener

- void changedUpdate(DocumentEvent e)
 (set)

- void insertUpdate(DocumentEvent e)
가가

- void changedUpdate(DocumentEvent e)
가

(validation)

가

,

가

가

0-9

JTextField

IntTextField

(validating)

(key)

(listener)

가

(consume) . ((consuming) 8 .)

3 3-3 (valid)

. look and feel (cut),
(copy), (paste) . metal look and feel
CTRL+V
가 가 .

Document.insertString (caret)
가 .)
PlainDocument . TextField
Document
IntTextDocument
PlainDocument insertString
insertString
(가)
IntTextDocument insertString
Integer.parseInt
super.insertString
PlainDocument.insertString
가 Integer.parseInt
IntTextDocument insertString
가 . (11
)

```
class IntTextDocument extends PlainDocument
{ public void insertString(int offs, String str,
  AttributeSet a)
  throws BadLocationException
{ if (str == null) return;
  String oldString = getText(0, getLength());
  String newString = oldString.substring(0, offs)
    + str + oldString.substring(offs);
  try
  { Integer.parseInt(newString + "0");
    super.insertString(offs, str, a);
  }
  catch(NumberFormatException e)
  {
  }
}
```

JTextField

IntTextField .

```
createDefaultModel      IntTextDocument  PlainDocument      .
```

```
class IntTextField extends JTextField
{ public IntTextField(int defval, int size)
  { super("" + defval, size);
  }

  protected Document createDefaultModel()
  { return new IntTextDocument();
  }
}
```

```
(entry) . 가
. -3 가 . -
. insertString( )
. 0 . -
. IntTextField
isValid
insertString isValid IntTextField getValue
. getValue
. isValid
.
-----
VB : KeyAscii 0
. VB java
가
-----
9-2 IntTextField 가 가
.
-----
: 가 가 가
Toolkit (beep)
Toolkit.getDefaultToolkit( ).beep( );
-----
```

9-2 ValidationText.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.text.*;

class ValidationTestFrame extends JFrame
  implements DocumentListener
{ public ValidationTestFrame()
```

```

{ setTitle("ValidationTest");
  setSize(300, 200);
  addWindowListener(new WindowAdapter()
  { public void windowClosing(WindowEvent e)
    { System.exit(0);
    }
  } );

  Container contentPane = getContentPane();

  JPanel p = new JPanel();
  hourField = new IntTextField(12, 3);
  p.add(hourField);
  hourField.getDocument().addDocumentListener(this);

  minuteField = new IntTextField(0, 3);
  p.add(minuteField);
  minuteField.getDocument().addDocumentListener(this);

  contentPane.add(p, "South");
  clock = new ClockPanel();
  contentPane.add(clock, "Center");
}

public void insertUpdate(DocumentEvent e)
{ setClock();
}
public void removeUpdate(DocumentEvent e)
{ setClock();
}
public void changedUpdate(DocumentEvent e)
{
}

public void setClock()
{ if (hourField.isValid() && minuteField.isValid())
  { int hours = hourField.getValue();
    int minutes = minuteField.getValue();
    clock.setTime(hours, minutes);
  }
}

private IntTextField hourField;
private IntTextField minuteField;
private ClockPanel clock;
}

class ClockPanel extends JPanel
{ public void paintComponent(Graphics g)
  { super.paintComponent(g);
    g.drawOval(0, 0, 100, 100);
    double hourAngle
      = 2 * Math.PI * (minutes - 3 * 60) / (12 * 60);
    double minuteAngle

```

```

        = 2 * Math.PI * (minutes - 15) / 60;
    g.drawLine(50, 50,
        50 + (int)(30 * Math.cos(hourAngle)),
        50 + (int)(30 * Math.sin(hourAngle)));
    g.drawLine(50, 50,
        50 + (int)(45 * Math.cos(minuteAngle)),
        50 + (int)(45 * Math.sin(minuteAngle)));
}

public void setTime(int h, int m)
{ minutes = h * 60 + m;
  repaint();
}

public void tick()
{ minutes++;
  repaint();
}

private int minutes = 0;
}

public class ValidationTest
{ public static void main(String[] args)
  { JFrame frame = new ValidationTestFrame();
    frame.show();
  }
}

class IntTextDocument extends PlainDocument
{ public void insertString(int offs, String str,
    AttributeSet a)
    throws BadLocationException
  { if (str == null) return;
    String oldString = getText(0, getLength());
    String newString = oldString.substring(0, offs)
        + str + oldString.substring(offs);
    try
    { Integer.parseInt(newString + "0");
      super.insertString(offs, str, a);
    }
    catch(NumberFormatException e)
    {
    }
  }
}

class IntTextField extends JTextField
{ public IntTextField(int defval, int size)
  { super("" + defval, size);
  }

  protected Document createDefaultModel()
  { return new IntTextDocument();
  }
}

```



```

가
    (*)
    JPasswordField

- -

API)
java.swing.JPasswordField
● JPasswordField(String text, int columns)
    :
    text null
    columns

● void setEchoChar(char echo)
    look and
feel
    0
    :
    echo 가

● char[] getPassword
    (overwrite)
    가 (garbage) 가

(Text Area)

가 JTextArea

ENTER 가 12 '\n' 가
StringTokenizer
9-12
JTextArea

textArea = new JTextArea(8, 40) // 8 40
getContentPane().add(textArea);

가 1 2
가
.setColumns
.setRows

(line-wrapping)
가
가

textArea.setLineWrap(true); //

```

9-12:

```
(wrapping) . - '\n'
.
가 .
(scroll pane) .
textArea = new JTextArea(8, 40);
JScrollPane scrollPane = new JScrollPane(textArea);
getContentPane().add(scrollPane, "Center");

가 . 가
가 .
.-
9-3
(line-wrapping) "wrap" "no wrap" "Insert"
(highlight) (section)
(paste) CTRL+X, CTRL+C CTRL+V (cut), (copy),
look and feel . (Metal, Windows, Mac look
and feel .)

-----
: JTextArea
(HTML RTF ) JEditorPane JTextPane
. 2
-----
```

9-3: TextAreaText.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TextAreaFrame extends JFrame
    implements ActionListener
{
    public TextAreaFrame()
    {
        JPanel p = new JPanel();

        insertButton = new JButton("Insert");
        p.add(insertButton);
        insertButton.addActionListener(this);

        wrapButton = new JButton("Wrap");
        p.add(wrapButton);
        wrapButton.addActionListener(this);

        noWrapButton = new JButton("No wrap");
        p.add(noWrapButton);
```

```

        noWrapButton.addActionListener(this);

        getContentPane().add(p, "South");

        textArea = new JTextArea(8, 40);
        scrollPane = new JScrollPane(textArea);
        getContentPane().add(scrollPane, "Center");

        setTitle("TextAreaTest");
        setSize(300, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }

    public void actionPerformed(ActionEvent evt)
    {
        Object source = evt.getSource();
        if (source == insertButton)
            textArea.append
                ("The quick brown fox jumps over the lazy dog. ");
        else if (source == wrapButton)
        {
            textArea.setLineWrap(true);
            scrollPane.validate();
        }
        else if (source == noWrapButton)
        {
            textArea.setLineWrap(false);
            scrollPane.validate();
        }
    }

    private JButton insertButton;
    private JButton wrapButton;
    private JButton noWrapButton;
    private JTextArea textArea;
    private JScrollPane scrollPane;
}

public class TextAreaTest {
    public static void main(String[] args)
    {
        JFrame f = new TextAreaFrame();
        f.show();
    }
}

API)
java.swing.JTextArea
● JTextArea(int rows, int cols)
    :
    rows
    cols

```

- `JTextArea(String text, int rows, int cols)`
 :
 text
 rows
 cols
 - `void setColumns(int cols)`
 :
 cols
 - `void setRows(int rows)`
 :
 cols
 - `void append(String newText)`
 :
 newText
 - `void setLineWrap(boolean wrap)`
 - (line-wrapping)
 wrap (wrap) true
 - `void setWrapStyleWord(boolean word)`
 word가 true word (boundary) (wrap) false
 word
- API)
 javax.swing.JScrollPane
- `JScrollPane(Component c)`
 가
 가
 :
 c
 - ((boundary) 가
 (identifier)가 (identify)
 - `JLabel`
 가
 - `JLabel`
 SwingConstants LEFT,

RIGHT, CENTER, NORTH, EAST
JLabel

```
JLabel label = new JLabel ("Text", SwingConstants.LEFT);
```

```
JLabel label = new JLabel("Text", JLabel.LEFT)
```

setText setIcon

9-13

가 "with"

API)

javax.swing.JLabel

- JLabel(String text)

:
text

- JLabel(Icon icon)

icon

- JLabel(String text, int align)

:
text
align SwingConstants.LEFT, SwingConstants.RIGHT,
SwingConstants.CENTER

- JLabel(String text, Icon icon, int align)

가 가
:
text
icon
align SwingConstants.LEFT, SwingConstants.RIGHT,
SwingConstants.CENTER

- void setText(String text)

:
text

- void setIcon(String text)

:
text

()

```

        가      . ,      가      가      가
        .

        selectAll()      가      ,      .

        .      ,
        .

select      . select      substring
    가      10      14
        , t.select(10, 15)      10      14
        .      (End-of-line)      .

getSelectionStart      getSelectionEnd
getSelectedText      .
        . Windows      SHIFT + arrow      .

API)
java.awt.text.JTextComponent
● void selectAll()
        .

● void select(int selStart, int selEnd)
        .
        :
        selStart
        selEnd      + 1

● int getSelectionStart()
        .

● int getSelectionEnd()
        + 1      .

● String getSelectedText()
        .

JTextArea
        가      ,      .
        .

        9-4      "find-and-replace"      7-
6      "Replace"

        .
        URL      .

```

9-4: TextEditTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TextEditFrame extends JFrame
{
    public TextEditFrame()
    {
        setTitle("TextEditTest");
        setSize(300, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        Container contentPane = getContentPane();

        JPanel panel = new JPanel();

        JButton replaceButton = new JButton("Replace");
        panel.add(replaceButton);
        replaceButton.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent evt)
            {
                String f = from.getText();
                int n = textArea.getText().indexOf(f);
                if (n >= 0 && f.length() > 0)
                    textArea.replaceRange(to.getText(), n,
                                            n + f.length());
            }
        });

        from = new JTextField(8);
        panel.add(from);

        panel.add(new JLabel("with"));

        to = new JTextField(8);
        panel.add(to);

        textArea = new JTextArea(8, 40);
        scrollPane = new JScrollPane(textArea);

        contentPane.add(panel, "South");
        contentPane.add(scrollPane, "Center");
    }

    private JScrollPane scrollPane;
    private JTextArea textArea;
    private JTextField from, to;
}

public class TextEditTest {
    public static void main(String[] args)
    {
        JFrame f = new TextEditFrame();
        f.show();
    }
}

```

```

    }
}

```

API)

java.awt.JTextArea

- void insertText(String str, int pos)

```

        :
        str
        pos          (0=          , '\n'          .)

```

- void replaceRange(String str, int start, int end)

```

        :
        str
        start
        end          + 1

```

' 가 .

(Check Box)

"yes" "no" ,

. 9-14 가 :

"italic"

(helper)

```

public CheckBoxFrame()

```

```

{   JPanel p = new JPanel();
    bold = addCheckBox(p, "Bold");
    italic = addCheckBox(p, "Italic");
    add(p, "South");
    . . .
}

```

```

public JCheckBox addCheckBox(JPanel p, String name)

```

```

{   JCheckBox c = new JCheckBox(name);
    c.addActionListener(this);
    p.add(c);
    return c;
}

```

9-14

```
setSelected  
.  
bold.setSelected(true);  
  
가  
actionPerformed  
.  
isSelected  
가 false true  
가  
가  
가
```

```
public void actionPerformed(ActionEvent evt)  
{ int m = (bold.isSelected() ? Font.BOLD : 0)  
  + (italic.isSelected() ? Font.ITALIC : 0);  
  panel.setFont(m);  
}
```

9-5

```
-----  
: AWT JCheckBox 가 Checkbox  
가  
-----
```

Example 9-5 CheckBoxTest.java

```
import java.awt.*;  
import java.awt.event.*;  
import javax.swing.*;  
  
class CheckBoxFrame extends JFrame  
  implements ActionListener  
{ public CheckBoxFrame()  
  { setTitle("CheckBoxTest");  
    setSize(300, 200);  
    addWindowListener(new WindowAdapter()  
      { public void windowClosing(WindowEvent e)  
        { System.exit(0);  
        }  
    } );  
  
    JPanel p = new JPanel();  
    bold = addCheckBox(p, "Bold");  
    italic = addCheckBox(p, "Italic");  
    getContentPane().add(p, "South");  
    panel = new CheckBoxTestPanel();  
    getContentPane().add(panel, "Center");  
  }  
  
  public JCheckBox addCheckBox(JPanel p, String name)  
  { JCheckBox c = new JCheckBox(name);
```

```

        c.addActionListener(this);
        p.add(c);
        return c;
    }

    public void actionPerformed(ActionEvent evt)
    {   int m = (bold.isSelected() ? Font.BOLD : 0)
        + (italic.isSelected() ? Font.ITALIC : 0);
        panel.setFont(m);
    }

    private CheckBoxTestPanel panel;
    private JCheckBox bold;
    private JCheckBox italic;
}

class CheckBoxTestPanel extends JPanel
{   public CheckBoxTestPanel()
    {   setFont(Font.PLAIN);
    }

    public void setFont(int m)
    {   setFont(new Font("SansSerif", m, 12));
        repaint();
    }

    public void paintComponent(Graphics g)
    {   super.paintComponent(g);
        g.drawString
            ("The quick brown fox jumps over the lazy dog.",
            0, 50);
    }
}

public class CheckBoxTest
{   public static void main(String[] args)
    {   JFrame frame = new CheckBoxFrame();
        frame.show();
    }
}

API)
java.awt.JCheckbox
●   JCheckbox(String label)
    :
    label

●   JCheckbox(String label, boolean state)
    :
    label
    state

●   JCheckbox(String label, boolean state)
    .

```

```

        :
        label
        icon

```

- boolean isSelected()
- void setSelected(boolean state)

```

가
button group)
Medium, Large, Extra-large
JRadioButton

가
(button group)
9-15
- Small,
ButtonGroup
JRadioButton

```

```

small = new JRadioButton("Small", false)
medium = new JRadioButton("Medium", true)
. . .
ButtonGroup group = new ButtonGroup ( );
group.add(small);
group.add(medium);
. . .

```

9-15

```

false
가 true
JPanel

```

9-14

9-15

가

```

public void actionPerformed(ActionEvent evt)
{
    Object source = evt.getSource();
    if(source == smallButton)
        panel.setSize(8);
    else if (source == mediumButton)
        panel.setSize(12);
    . . .
}

```

```

        isSelected
        .
        가
        .
        ButtonGroup 가
        . ButtonGroup getSelection
        가
        ButtonModel
        .
        (view) 가
        . ButtonModel
        ItemSelectable . ButtonModel
        getSelectedObjects
        null
        . ButtonModel
        getActionCommand
        . setActionCommand 가
        .

```

```
buttonGroup.getSelection().getActionCommand( )
```

9-7

BorderText

9-6

9-6:RadioButtonTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class RadioButtonFrame extends JFrame
    implements ActionListener
{
    public RadioButtonFrame()
    {
        setTitle("RadioButtonTest");
        setSize(400, 200);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }

    JPanel buttonPanel = new JPanel();
    ButtonGroup group = new ButtonGroup();
    smallButton =
        addRadioButton(buttonPanel, group, "Small", false);
    mediumButton =
        addRadioButton(buttonPanel, group, "Medium", true);
    largeButton =
        addRadioButton(buttonPanel, group, "Large", false);
    xlargeButton =
        addRadioButton(buttonPanel, group, "Extra large",
            false);
}

```

```

        getContentPane().add(buttonPanel, "South");
        panel = new RadioButtonTestPanel();
        getContentPane().add(panel, "Center");
    }

    public JRadioButton addRadioButton(JPanel buttonPanel,
        ButtonGroup g, String buttonName, boolean v)
    { JRadioButton button = new JRadioButton(buttonName, v);
      button.addActionListener(this);
      g.add(button);
      buttonPanel.add(button);
      return button;
    }

    public void actionPerformed(ActionEvent evt)
    { Object source = evt.getSource();
      if(source == smallButton)
          panel.setSize(8);
      else if (source == mediumButton)
          panel.setSize(12);
      else if (source == largeButton)
          panel.setSize(14);
      else if (source == xlargeButton)
          panel.setSize(18);
    }

    private RadioButtonTestPanel panel;
    private JRadioButton smallButton;
    private JRadioButton mediumButton;
    private JRadioButton largeButton;
    private JRadioButton xlargeButton;
}

class RadioButtonTestPanel extends JPanel
{ public RadioButtonTestPanel()
  { setSize(12);
  }

  public void setSize(int p)
  { setFont(new Font("SansSerif", Font.PLAIN, p));
    repaint();
  }

  public void paintComponent(Graphics g)
  { super.paintComponent(g);
    g.drawString
      ("The quick brown fox jumps over the lazy dog.",
       0, 50);
  }
}

public class RadioButtonTest
{ public static void main(String[] args)
  { JFrame frame = new RadioButtonFrame();
  }
}

```

```

        frame.show();
    }
}
API)
javax.swing.JRadioButton
JRadioButton(String label, boolean state)
:
    label
    state

```

```

JRadioButton(String label, Icon icon)
:
    label
    state

```

```

API)
javax.swing.ButtonGroup
● void add(AbstractButton b)
    가
● ButtonModel getSelection( )
    .

```

```

API)
javax.swing.ButtonModel
● String getActionCommand
    .

```

```

API)
javax.swing.AbstractButton
● void setActionCommand(String s)
    .

```

(Border)

JComponent . 가 .

1. BorderFactory .

```

Lowered bevel
Raised bevel
Etched
Line
Matte
Empty ( .)

```

2. .

```

BorderFactory.createTitledBorder
3. BorderFactory.createCompoundBorder
4. JComponent.setBorder

        etched

Border etched = BorderFactory.createEtchedBorder ( );
Border titled = BorderFactory.createTitledBorder (etched, "A Title");
panel.setBorder(titled);

9-7 가 (option) 가 API
n 가 beveled
border 가 SoftBevelBorder 가
        BorderFactory 가

```

9-16

9-7:BorderText.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;

class BorderFrame extends JFrame
    implements ActionListener
{
    public BorderFrame()
    {
        JPanel buttonPanel = new JPanel();
        group = new ButtonGroup();
        addRadioButton(buttonPanel, group, "Lowered bevel",
            true);
        addRadioButton(buttonPanel, group, "Raised bevel",
            false);
        addRadioButton(buttonPanel, group, "Etched",
            false);
        addRadioButton(buttonPanel, group, "Line",
            false);
        addRadioButton(buttonPanel, group, "Matte",
            false);
        addRadioButton(buttonPanel, group, "Empty",
            false);

        Border etched = BorderFactory.createEtchedBorder();
        Border titled = BorderFactory.createTitledBorder
            (etched, "Border types");
        buttonPanel.setBorder(titled);

        getContentPane().add(buttonPanel, "South");

        setDemoPanel();

        setTitle("BorderTest");
        setSize(600, 200);
    }
}

```

```

        addWindowListener(new WindowAdapter()
        { public void windowClosing(WindowEvent e)
          { System.exit(0);
          }
        } );
    }

    public void addRadioButton(JPanel buttonPanel,
        ButtonGroup g, String buttonName, boolean v)
    { JRadioButton button = new JRadioButton(buttonName, v);
      button.addActionListener(this);
      g.add(button);
      buttonPanel.add(button);
      button.setActionCommand(buttonName);
    }

    public void actionPerformed(ActionEvent evt)
    { setDemoPanel();
    }

    public void setDemoPanel()
    { JPanel panel = new JPanel();
      Border border = null;
      String command = group.getSelection()
          .getActionCommand();
      if (command.equals("Lowered bevel"))
          border = BorderFactory.createLoweredBevelBorder();
      else if (command.equals("Raised bevel"))
          border = BorderFactory.createRaisedBevelBorder();
      else if (command.equals("Etched"))
          border = BorderFactory.createEtchedBorder();
      else if (command.equals("Line"))
          border
              = BorderFactory.createLineBorder(Color.blue);
      else if (command.equals("Matte"))
          border = BorderFactory.createMatteBorder(10, 10,
              10, 10, Color.blue);
      else if (command.equals("Empty"))
          border = BorderFactory.createEmptyBorder();

      panel.setBorder(border);
      getContentPane().add(panel, "Center");
      validate();
    }

    private JPanel panel;
    private ButtonGroup group;
}

public class BorderTest
{ public static void main(String[] args)
  { JFrame frame = new BorderFrame();
    frame.show();
  }
}

```

```
}
```

API)

```
java.swing.BorderFactory
```

- static Border createLineBorder(Color color)
- static Border createLineBorder(Color color, int thickness)

- static MatteBorder createMatteBorder(int top, int left, int bottom, int right, Color color)
- static MatteBorder createMatteBorder(int top, int left, int bottom, int right, Icon tileIcon)
- border

- static Border createEmptyBorder ()
- static Border createEmptyBorder (int top, int left, int bottom, int right)

- static Border createEtchedBorder ()
- static Border createEtchedBorder (Color highlight, Color shadow)
- 3 가

- highlight, shadow 3
- static Border createBevelBorder ()
- static Border createBevelBorder (int type, Color highlight, Color shadow)
- static Border createLowerdBevelBorder ()
- static Border createRaisedBevelBorder ()
- 가
- :
- type BevelBorder.LOWERED, BevelBorder.RAISED
- highlight , shadow 3

- static TitledBorder CreateTitledBordeer (String title)
- static TitledBorder CreateTitledBordeer (Border border)
- static TitledBorder CreateTitledBordeer (Border border, String title)
- static TitledBorder CreateTitledBordeer (Border border, String title, int justification, int position)
- static TitledBorder CreateTitledBordeer (Border border, String title, int justification, int position, Font font)
- static TitledBorder CreateTitledBordeer (Border border, int justification, int position, Font font, Color)
- :
- title
- border
- justification TitleBorder.LEFT, TitleBorder.RIGHT,
- TitleBorder.CENTER
- position ABOVE_TOP, TOP, BELOW_TOP, ABOVE_BOTTOM,
- BOTTOM, BELOW_BOTTOM
- font
- color

- static CompoundBorder createCompoundBorder(Border outsideBorder, Border insideBorder)

API)

javax.swing.border.SoftBevelBorder

- SoftBevelBorder(int type)
- SoftBevelBorder(int type, Color highlight, Color shadow)

```

        가 bevel
        :
        type BevelBorder.LOWERED, BevelBorder.RAISED
        color, shadow 3D

```

API)

javax.swing.JComponent

- void setBorder(Border border)

```

        . , 가
        가
        . 가
        가
        (machinery) 가

```

JList

JList

```

        가
        .
        9-17 , "quick",
        "quickly", "brown", "hungry", "wild", "static", "final", "private"

```

JList

```

String[] words = {"quickly", "brown", "hungry", "wild", . . .};
JList wordList = new JList(words);

```

9-17

```

        (anonymous)
JList wordList = new JList (new String[ ]{"quickly", "brown", "hungry",
"wild", . . .});

```

```
JScrollPane scrollPane = new JScrollPane(wordList);
```

가

가

가

8 가

```
. setVisibleRowCount
```

```
wordList.setVisibleRowCount(10); // 10
```

CTRL

SHIFT

```
setSelectionMode 가
```

```
wordList.setSelectionMode
(ListSelectionMode.SINGLE_SELECTION);
//
```

```
wordList.setSelectionMode
(ListSelectionMode.SINGLE_INTERVAL_SELECTION);
//
```

가

```
public void valueChanged(ListSelectionEvent evt)
```

```
(listener)
```

```
class ListFrame extends JFrame
implements ListSelectionListener
```

```
{ public ListFrame()
```

```
{
```

```
    JList wordList = new JList(words);
    wordList.addListSelectionListener(this);
```

```
}
```

```
public void valueChanged(ListSelectionEvent evt)
```

```
{
```

```
}
```

```
}
```

가

가

가

2

가

가

(transitional)

```

evt.isAdjusting( )

isAdjusting      가      가      final      true      .
      (transitional)      isAdjusting      false
      false 가      가      isAdjusting
      (feedback)      가      .

-----
      ;      (CTRL+ )
isAdjusting      true      가      가      가
      isAdjusting
-----

      가
getSelectedValues

-----
VB      : getSelectedValues      VB      List property
-----

      (cast)      가      .

JList source = (JList)evt.getSource();
Object[] values = source.getSelectedValues();
for (int i = 0; i < values.length; i++);

(String)values[i]      가      .

-----
      : Object []      getSelectedValues      String[ ]
      가      .

int length = values.length;
String[ ] words = new String(length);
System.arraycopy(values, 0, words, 0 , length);
-----

      getSelectedValue
      . (
      .)

String selection = (String)source.getSelectedValue ( );

-----
      :
      가

```

```

public void mouseClicked(MouseEvent evt)
{
    if (evt.getClickCount ( ) == 2)
    {
        JList source = (JList)evt.getSource();
        Object[ ] selection = source.getSelectedValues( );
        doAction(selection);
    }
}

```

9-8 . valueChanged 가

9-8:ListTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

class ListFrame extends JFrame
    implements ListSelectionListener
{
    public ListFrame()
    {
        setTitle("ListTest");
        setSize(400,300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        String[] words =
        { "quick","brown","hungry","wild","silent",
          "huge","private","abstract","static","final"
        };

        JList wordList = new JList(words);
        JScrollPane scrollPane = new JScrollPane(wordList);

        JPanel p = new JPanel();
        p.add(scrollPane);
        wordList.addListSelectionListener(this);

        getContentPane().add(p, "South");
        panel = new ListTestPanel();
        getContentPane().add(panel, "Center");
    }

    public void valueChanged(ListSelectionEvent evt)
    {
        JList source = (JList)evt.getSource();
        Object[] values = source.getSelectedValues();

        String text = "";
        for (int i = 0; i < values.length; i++)
        {
            String word = (String)values[i];

```

```

        text += word + " ";
    }

    panel.setAttribute(text);
}

private ListTestPanel panel;
}

class ListTestPanel extends JPanel
{
    public ListTestPanel()
    {
        setAttribute("");
    }

    public void setAttribute(String w)
    {
        text = "The " + w + "fox jumps over the lazy dog.";
        repaint();
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.drawString(text, 0, 50);
    }

    private String text;
}

public class ListTest
{
    public static void main(String[] args)
    {
        JFrame frame = new ListFrame();
        frame.show();
    }
}

API)
javax.swing.JList
● JList(Object[] items)

.

● void setVisibleRowCount(int c)

.

● void setVisibleRowCount (int c)

.

:
mode SINGLE_SELECTION, SINGLE_INTERVAL_SELECTION,
MULTIPLE_INTERVAL_SELECTION

● void addListSelectionListener(ListSelectionListener listener)
가
● Object [] getSelectedValue( )
.
● Object getSelectedValue( )
null

```

API)
 javax.swing.event.ListSelectionListener
 ● void valueChanged(ListSelectionEvent e)

(list model)

가

(collection) 가 JList

가 ? JList

() () ()

JList 가

ListModel

가

```
public interface ListModel
{
    public int getSize();
    public Object getElementAt(int n);
    public void addListDataListener(ListDataListener l);
    public void removeListDataListener(ListDataListener l);
}
```

JList JList list data listener

? JList (vector)

? getElementAt
 가 가 가

3

9-18:

26*26*26=17576
가

3

.

.

AbstractListModel

가

getSize getElementAt

```
class WordListModel extends AbstractListModel
{ public WordListModel(int n) { length = n; }
  public int getSize() { return (int)Math.pow(26, length); }
  public Object getElementAt(int n)
  {
    . . .
  }
  . . .
}
```

n

(technical)

9-8

```
JList wordList = new JList(new WordListModel(3));
wordList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
JScrollPane scrollPane = new JScrollPane(wordList);
```

가

가

가

가

```
wordList.setFixedCellWidth(50);
wordList.setFixedCellHeight(15);
```

가

9-9

가

가

가
(entity)

9-9: LonglistTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

class WordListModel extends AbstractListModel
{ public WordListModel(int n) { length = n; }
  public int getSize()
```

```

    { return (int)Math.pow(LAST - FIRST + 1, length);
    }
    public Object getElementAt(int n)
    { String r = "";
      for (int i = 0; i < length; i++)
      { char c = (char)(FIRST + n % (LAST - FIRST + 1));
        r = c + r;
        n = n / (LAST - FIRST + 1);
      }
      return r;
    }

    private int length;
    public static final char FIRST = 'a';
    public static final char LAST = 'z';
}

class LongListFrame extends JFrame
    implements ListSelectionListener
{ public LongListFrame()
  { setTitle("LongListTest");
    setSize(400, 300);
    addWindowListener(new WindowAdapter()
      { public void windowClosing(WindowEvent e)
        { System.exit(0);
        }
      } );

    JList wordList = new JList(new WordListModel(3));
    wordList.setSelectionMode
      (ListSelectionModel.SINGLE_SELECTION);

    wordList.setFixedCellWidth(50);
    wordList.setFixedCellHeight(15);

    JScrollPane scrollPane = new JScrollPane(wordList);

    JPanel p = new JPanel();
    p.add(scrollPane);
    wordList.addListSelectionListener(this);

    getContentPane().add(p, "South");
    panel = new LongListPanel();
    getContentPane().add(panel, "Center");
  }

  public void valueChanged(ListSelectionEvent evt)
  { JList source = (JList)evt.getSource();
    String word = (String)source.getSelectedValue();
    panel.setJumper(word);
  }

  private LongListPanel panel;
}

```

```

class LongListPanel extends JPanel
{
    public LongListPanel()
    {
        setJumper("fox");
    }

    public void setJumper(String w)
    {
        text = "The quick brown "
            + w + " jumps over the lazy dog.";
        repaint();
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.drawString(text, 0, 50);
    }

    private String text;
}

public class LongListTest
{
    public static void main(String[] args)
    {
        JFrame frame = new LongListFrame();
        frame.show();
    }
}

```

API)

javax.swing.JList

- JList(ListModel, dataModel)

- void setFixedCellWidth(int width)
가 0 . -1
가 .

- void setFixedCellHeight(int height)
가 0 . -1
.

API)

- int getSize()

- Object getElementAt(int index)

가

model

가

ListModel model = list.getModel ();

ListModel

```

        가 .
        JList

Vector values = new Vector ( );
values.addElement("quick");
values.addElement("brown");
. . .
JList list = new JList(values);

(vector) .
.
.
DefaultListModel .

DefalutListModel model = new DefaultListModel ( );
model.addElement("quick");
model.addElement("brown");
. . .
JList list = new JList(values);
model

model.removeElement("quick");
model.addElement("slow");

DefaultModel Vector
. (
add remove DefaultListModel .)
AbstractListModel

-----
: JList 가
. 가 DefaultListModel
. - 가
. Vector JList
가 .

public JList(final Vector listData)
{ this (new AbstractListModel()
{ public int getSize( ) { return listData.size();}
public Object getElementAt(int i) { return
listData.elementAt(i); }
});
}

가

( final (vector)
listData
. listData 가 .)

```

API)

javax.swing.JList

- listModel getModel ()

.

API)

javax.swing.DefaultListModel

- void addElement(Object obj)

가 .

- boolean removeElement (Object obj)

(occurence)

. 가

true

false

.

-Icon

(drawing)

가

JList 가

(list cell renderer) JList

가 .

interface ListCellRenderer

```
{ Component getListCellRendererComponent(JList list,
    Object value, int index,
    boolean isSelected, boolean cellHasFocus);
}
```

.

Component

getPreferredSize

.

2

가

.

class MyCellRenderer implements ListCellRenderer

```
{ public Component getListCellRendererComponent (final JList list,
    final Object value, final int index,
    final boolean isSelected, final boolean cellHasFocus)
{ return new JPanel()
    { public void paintComponent(Graphics g)
        { //
        }

        public Dimension getPreferredSize()
        { //
        }
    };
}
```

}

9-10

```
. paintComponent
    JList look and feel
가 . JList getForeground /getBackground
getSelectionForeground/getSelectionBackground
    . 7 getPreferredSize
가 .
```

9-19

가

가 .

```
class FontCellRenderer implements ListCellRenderer
{ public Component getListCellRendererComponent
  (final JList list, final Object value,
   final int index, final boolean isSelected,
   final boolean cellHasFocus)
{ return new JPanel()
  { public void paintComponent(Graphics g)
    { Font font = (Font)value;
      String text = font.getFamily();
      FontMetrics fm = g.getFontMetrics(font);
      g.setColor(isSelected
        ? list.getSelectionBackground()
        : list.getBackground());
      g.fillRect(0, 0, getWidth(), getHeight());
      g.setColor(isSelected
        ? list.getSelectionForeground()
        : list.getForeground());
      g.setFont(font);
      g.drawString(text, 0, fm.getAscent());
    }

    public Dimension getPreferredSize()
    { Font font = (Font)value;
      String text = font.getFamily();
      Graphics g = getGraphics();
      FontMetrics fm = g.getFontMetrics(font);
      return new Dimension(fm.stringWidth(text),
        fm.getHeight());
    }
  };
}
```

} setCellRenderer .

fontList.setCellRenderer(new FontCellRenderer());

(custom render) .

가

가 . JLabel

```

class FontCellRenderer implements ListCellRenderer
{
    public Component getListCellRendererComponent
        (JList list, Object value, int index, boolean isSelected, boolean
        cellHasFocus)
    {
        JLabel label = new JLabel( );
        Font font = (Font)value;
        label.setText = (font.getFamily());
        label.setFont(font);
        label.setOpaque(true);
        label.setBackground(isSelected
            ? list.getSelectionBackground()
            : list.getBackground());
        label.setForeground(isSelected
            ? list.getSelectionForeground()
            : list.getForeground());
        return label;
    }
}

```

```

JLabel paintComponent    getPreferredSize
FontCellRenderere JLabel
getListCellRenderComponent
this

```

```

class FontCellRenderer implements ListCellRenderer
{
    public Component getListCellRendererComponent
        (JList list, Object value, int index, boolean isSelected, boolean
        cellHasFocus);
    {
        JLabel label = new JLabel( );
        Font font = (Font)value;
        label.setText = (font.getFamily());
        label.setFont(font);
        label.setOpaque(true);
        label.setBackground(isSelected
            ? list.getSelectionBackground()
            : list.getBackground());
        label.setForeground(isSelected
            ? list.getSelectionForeground()
            : list.getForeground());
        return this;
    }
}

```

- JLabel-가

9-10:ListRenderingTest.java

```

import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

class FontCellRenderer implements ListCellRenderer
{
    public Component getListCellRendererComponent
        (final JList list, final Object value,
         final int index, final boolean isSelected,
         final boolean cellHasFocus)
    {
        return new JPanel()
        {
            public void paintComponent(Graphics g)
            {
                Font font = (Font)value;
                String text = font.getFamily();
                FontMetrics fm = g.getFontMetrics(font);
                g.setColor(isSelected
                    ? list.getSelectionBackground()
                    : list.getBackground());
                g.fillRect(0, 0, getWidth(), getHeight());
                g.setColor(isSelected
                    ? list.getSelectionForeground()
                    : list.getForeground());
                g.setFont(font);
                g.drawString(text, 0, fm.getAscent());
            }

            public Dimension getPreferredSize()
            {
                Font font = (Font)value;
                String text = font.getFamily();
                Graphics g = getGraphics();
                FontMetrics fm = g.getFontMetrics(font);
                return new Dimension(fm.stringWidth(text),
                                     fm.getHeight());
            }
        };
    }
}

class ListRenderingFrame extends JFrame
    implements ListSelectionListener
{
    public ListRenderingFrame()
    {
        setTitle("ListRenderingTest");
        setSize(400, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }

    Vector fonts = new Vector();
    fonts.add(new Font("Serif", Font.PLAIN, 12));
    fonts.add(new Font("SansSerif", Font.PLAIN, 12));
}

```

```

        fonts.add(new Font("Monospaced", Font.PLAIN, 12));
        fonts.add(new Font("Dialog", Font.PLAIN, 12));
        fonts.add(new Font("DialogInput", Font.PLAIN, 12));
        JList fontList = new JList(fonts);
        fontList.setSelectionMode
            (ListSelectionModel.SINGLE_SELECTION);
        fontList.setCellRenderer(new FontCellRenderer());
        JScrollPane scrollPane = new JScrollPane(fontList);

        JPanel p = new JPanel();
        p.add(scrollPane);
        fontList.addListSelectionListener(this);

        getContentPane().add(p, "South");
        panel = new ListRenderingPanel();
        getContentPane().add(panel, "Center");
    }

    public void valueChanged(ListSelectionEvent evt)
    {
        JList source = (JList)evt.getSource();
        Font font = (Font)source.getSelectedValue();
        panel.setFont(font);
    }

    private ListRenderingPanel panel;
}

class ListRenderingPanel extends JPanel
{
    public ListRenderingPanel()
    {
        setFont(new Font("Serif", Font.PLAIN, 12));
    }

    public void setFont(Font f)
    {
        currentFont = f;
        repaint();
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.setFont(currentFont);
        g.drawString
            ("The quick brown fox jumps over the lazy dog",
             0, 50);
    }

    private Font currentFont;
}

public class ListRenderingTest
{
    public static void main(String[] args)
    {
        JFrame frame = new ListRenderingFrame();
        frame.show();
    }
}

```

API)

javax.swing.JList

- Color getBackground ()

.

- Color getSelectionBackground ()

.

- void setCellRender(ListCellRender cellrender)

.

API)

javax.swing.ListCellRenderer

- Component getList (JList list, Object item, int index, boolean isSelected, boolean HasFocus)

paint 가 cell

getPreferedSize

.

:

list

item

isSelected true

hasFocus true

가

. 가

가

.

.

9-20

가 (editable)

.

(combo

box)

가

. setEditable

.

JComboBox

.

getSelectedItem

가

.

(Serif, SansSerif,

Monospaced, etc)

가

.

addItem

.

addItem

.

style = new JComboBox();

style.setEditable(true);

style.addItem("Serif");

style.addItem("SansSerif");

.

insertItemAt

```
style.insetItemAt("Monospaced", 0)
```

```
removeItem removeItemAt
```

```
style.removeItem("Monospaced")
style.removeItemAt(0)
```

```
RemoveAllItem
```

```
가
```

```
getActionCommand
```

```
getSelectedItem
```

```
getSource
```

```
public void actionPerformed(ActionEvent evt)
{ JComboBox source = (JComboBox)evt.getSource();
  String item = (String)source.getSelectedItem();
  panel.setStyle(item);
}
```

```
9-11
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
```

```
class ComboBoxFrame extends JFrame
  implements ActionListener
{ public ComboBoxFrame()
  { setTitle("ComboBoxTest");
    setSize(300,200);
    addWindowListener(new WindowAdapter()
      { public void windowClosing(WindowEvent e)
        { System.exit(0);
        }
      }
    );
  }
```

```
    style = new JComboBox();
    style.setEditable(true);
    style.addItem("Serif");
    style.addItem("SansSerif");
    style.addItem("Monospaced");
    style.addItem("Dialog");
    style.addItem("DialogInput");
    style.addActionListener(this);
```

```
    JPanel p = new JPanel();
    p.add(style);
    getContentPane().add(p, "South");
    panel = new ComboBoxTestPanel();
    getContentPane().add(panel, "Center");
}
```

```
public void actionPerformed(ActionEvent evt)
```

```

    { JComboBox source = (JComboBox)evt.getSource();
      String item = (String)source.getSelectedItem();
      panel.setStyle(item);
    }

    private ComboBoxTestPanel panel;
    private JComboBox style;
}

class ComboBoxTestPanel extends JPanel
{ public ComboBoxTestPanel()
  { setStyle("Serif");
  }

  public void setStyle(String s)
  { setFont(new Font(s, Font.PLAIN, 12));
    repaint();
  }

  public void paintComponent(Graphics g)
  { super.paintComponent(g);
    g.drawString
      ("The quick brown fox jumps over the lazy dog.",
       0, 50);
  }
}

public class ComboBoxTest
{ public static void main(String[] args)
  { JFrame frame = new ComboBoxFrame();
    frame.show();
  }
}
API)
javax.swing.JComboBox
● void setEditable(boolean b)
    :
    b 가 true, false

● void addItem(Object item)
    .

● void insertItemAt(Object item, int index)
    .

● void removeItem(Object item)
    .

● void removeItemAt(int index)

● void removeAllItems( )
    .

```

- Object `getSelectedItem( )`

가

-
-

가

가 .

```

: (Adjustable.HORIZONTAL Adjustable.VERTICAL)
  setOrientation 가
.
: . getValue
  setValue .
: 0 100 . setMinimum
setMaximum .

```

(visible area):

가

가

(9-21)

0

setVisibleAmount

```

가 : 가 , 1
setLineIncrement .

```

```

가 (block increment): 가 ,
가 (block increment) 10 .
setLineIncrement setPageIncrement .

```

```

(Adjustable.HORIZONTAL, Adjustable.VERTICAL)
.. , (visible amount)
SetValues . SetValues

```

.

red, green, blue

. (9-22) 2 가

```

red = new JScrollBar(Adjustable, HORIZONTAL)
red.setValues(0, 0, 0, 255)
red = new JScrollBar(Adjustable.HORIZONTAL, 0 , 0, 0, 255)

```

가

(Adjustment) . 5 가


```

        0, 0, 0, 255));
red.setBlockIncrement(16);
red.addAdjustmentListener(this);

p.add(greenLabel = new JLabel("Green 0"));
p.add(green = new JScrollBar(Adjustable.HORIZONTAL,
    0, 0, 0, 255));
green.setBlockIncrement(16);
green.addAdjustmentListener(this);

p.add(blueLabel = new JLabel("Blue 0"));
p.add(blue = new JScrollBar(Adjustable.HORIZONTAL,
    0, 0, 0, 255));
blue.setBlockIncrement(16);
blue.addAdjustmentListener(this);

contentPane.add(p, "South");

colorPanel = new JPanel();
colorPanel.setBackground(new Color(0, 0, 0));
contentPane.add(colorPanel, "Center");
}

public void adjustmentValueChanged(AdjustmentEvent evt)
{
    redLabel.setText("Red " + red.getValue());
    greenLabel.setText("Green " + green.getValue());
    blueLabel.setText("Blue " + blue.getValue());
    colorPanel.setBackground(new Color(red.getValue(),
        green.getValue(), blue.getValue()));

    colorPanel.repaint();
}

public static void main(String[] args)
{
    Frame f = new ColorSelect();
    f.show();
}

private JLabel redLabel;
private JLabel greenLabel;
private JLabel blueLabel;

private JScrollBar red;
private JScrollBar green;
private JScrollBar blue;

private JPanel colorPanel;
}

API)
javax.swing.Scrollbar
● JScrollbar(int orientation)
    :
    orientation HORIZONTAL    VERTICAL

```

- JScrollbar(int orientation, int value, int visible, int minimum, int maximum)
 - : orientation HORIZONTAL VERTICAL
 - value
 - visible 0
 - minimum
 - maximum
- void setValue(int Value)
 - value ,
 - .
- void setValues(int value, int visible, int minimum, int maximum)
 - value
 - visible 0
 - minimum
 - maximum
- void setMinimum(int value)
 - :
 - value
- void setMaximum(int value)
 - :
 - value
- void setVisibleAmount(int value)
 - :
 - value (visible setting)
- void setBlockIncrement(int i)
 - 가 . 가 가
- void setUnitIncrement(int i)
 - 가 . 가 가
- int getvalue()

API)

java.awt.event.AdjustmentListener

- void adjustmentValueChanged(AdjustmentEvent e)

.

(Scroll Panes)

.

- - (offset)
 - (offset)
- ScrollPane
- ScrollPane
- 가
- ScrollPane
- 가
- 가

```

ScrollPane sp = new JScrollPane(viewdComponent);

//

ScrollPane sp = new JScrollPane(viewedComponent,
    ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
    ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER,
);

ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED
ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED 가

contentPane.add(sp, "Center")

가

public MyComponent( )
{ setPreferredSize(new Dimension(MAX_XWIDTH, MAX_YHEIGHT));
}

viewport
JViewport vp = sp.getViewport( );
vp.setViewSize(new Dimension(MAX_XWIDTH, MAX_YHEIGHT));

viewport viewport
가 (

```

```

9-12
        600*400
300*200
        MousePanel
        MousePanel
        MousePanel
        JScrollPane
        9-23
        (header)
        4
9-23
        ruler
        RulerPanel
        paintComponent
        9-24
        setColumnHeaderView
        (
        setRowHeaderView
RulerPanel horizRulerPanel = new RulerPanel
    (SwingConstants.HORIZONTAL,
    viewedComponent.getPreferredSize().width,
    25, 100, 100, 10);
sp.setColumnHeaderView(horizRulerPanel);
        setCorner
        ( SwingConstants.NORTH_WEST
        JScrollPane.UPPER_RIGHT_CORNER
        Swing
        4
sp.setCorner(JScrollPane.UPPER_RIGHT_CORNER, new JLabel(new
ImageIcon("blue-ball.gif")));
        PGUP/PGDN,
CTRL+PGUP/CTRL+PGDN
        8
        JComponent
        registerKeyboardAction
        Action
        ScrollAction
        (-1

```

```

+1      ,      )      . actionPerformed
JScrollBar      setVaule      .

```

```

class ScrollAction extends AbstractAction
{
public void actionPerformed(ActionEvent evt)
{ JScrollBar scrollBar;
  if (orientation == JScrollBar.HORIZONTAL)
    scrollBar = scrollPane.getHorizontalScrollBar();
  else
    scrollBar = scrollPane.getVerticalScrollBar();
  if (scrollBar == null || !scrollBar.isVisible())
    return;
  int increment;
  if (type == UNIT)
    increment = scrollBar.getUnitIncrement();
  else
    increment = scrollBar.getBlockIncrement();
  scrollBar.setValue(scrollBar.getValue() +
    increment * direction);
}
}

```

ScrollAction 가

```

sp.registerKeyboardAction(
    new ScrollAction(p, orient, t, dir),
    KeyStroke.getKeyStroke(key, modifier, false),
    JComponent.WHEN_IN_FOCUSED_WINDOW);

```

register ScrollAction

9-13 .

9-13: ScrollPaneTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class MousePanel extends JPanel
// unchanged except for setPreferredSize
implements MouseMotionListener
{ public MousePanel()
{ addMouseListener(new MouseAdapter()
{ public void mousePressed(MouseEvent evt)
{ int x = evt.getX();
  int y = evt.getY();
  current = find(x, y);
  if (current < 0) // not inside a square
    add(x, y);
}

public void mouseClicked(MouseEvent evt)
{ int x = evt.getX();

```

```

        int y = evt.getY();

        if (evt.getClickCount() >= 2)
        { remove(current);
        }
    }
});
addMouseMotionListener(this);

setPreferredSize(new Dimension(MAX_XWIDTH,
    MAX_YHEIGHT));
}

public void paintComponent(Graphics g)
{ super.paintComponent(g);
    for (int i = 0; i < nsquares; i++)
        draw(g, i);
}

public int find(int x, int y)
{ for (int i = 0; i < nsquares; i++)
    { if (squares[i].x - SQUARELENGTH / 2 <= x &&
        x <= squares[i].x + SQUARELENGTH / 2
        && squares[i].y - SQUARELENGTH / 2 <= y
        && y <= squares[i].y + SQUARELENGTH / 2)
        return i;
    }
    return -1;
}

public void draw(Graphics g, int i)
{ g.drawRect(squares[i].x - SQUARELENGTH / 2,
    squares[i].y - SQUARELENGTH / 2,
    SQUARELENGTH, SQUARELENGTH);
}

public void add(int x, int y)
{ if (nsquares < MAXNSQUARES)
    { squares[nsquares] = new Point(x, y);
      current = nsquares;
      nsquares++;
      repaint();
    }
}

public void remove(int n)
{ if (n < 0 || n >= nsquares) return;
  nsquares--;
  squares[n] = squares[nsquares];
  if (current == n) current = -1;
  repaint();
}

public void mouseMoved(MouseEvent evt)
{ int x = evt.getX();

```

```

        int y = evt.getY();

        if (find(x, y) >= 0)
            setCursor(Cursor.getPredefinedCursor(
                Cursor.CROSSHAIR_CURSOR));
        else
            setCursor(Cursor.getDefaultCursor());
    }

    public void mouseDragged(MouseEvent evt)
    {
        int x = evt.getX();
        int y = evt.getY();

        if (current >= 0)
        {
            Graphics g = getGraphics();
            g.setXORMode(getBackground());
            draw(g, current);
            squares[current].x = x;
            squares[current].y = y;
            draw(g, current);
            g.dispose();
        }
    }

    private static final int MAX_XWIDTH = 600;
    private static final int MAX_YHEIGHT = 400;
    private static final int SQUARELENGTH = 10;
    private static final int MAXNSQUARES = 100;
    private Point[] squares = new Point[MAXNSQUARES];
    private int nsquares = 0;
    private int current = -1;
}

class ScrollPaneFrame extends JFrame
{
    public ScrollPaneFrame()
    {
        setTitle("ScrollPaneTest");
        setSize(300, 200);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        Container contentPane = getContentPane();
        Component viewedComponent = new MousePanel();
        JScrollPane sp = new JScrollPane(viewedComponent);

        RulerPanel horizRulerPanel = new RulerPanel(
            SwingConstants.HORIZONTAL,
            viewedComponent.getPreferredSize().width,
            25, 100, 100, 10);
        sp.setColumnHeaderView(horizRulerPanel);

        RulerPanel vertRulerPanel = new RulerPanel

```

```

        (SwingConstants.VERTICAL,
         25, viewedComponent.getPreferredSize().height,
         100, 100, 10);
sp.setRowHeaderView(vertrulerPanel);
contentPane.add(sp, "Center");

ScrollAction.register(sp, JScrollBar.HORIZONTAL,
    ScrollAction.UNIT, -1, KeyEvent.VK_LEFT, 0);
ScrollAction.register(sp, JScrollBar.HORIZONTAL,
    ScrollAction.UNIT, 1, KeyEvent.VK_RIGHT, 0);
ScrollAction.register(sp, JScrollBar.VERTICAL,
    ScrollAction.UNIT, -1, KeyEvent.VK_UP, 0);
ScrollAction.register(sp, JScrollBar.VERTICAL,
    ScrollAction.UNIT, 1, KeyEvent.VK_DOWN, 0);
ScrollAction.register(sp, JScrollBar.HORIZONTAL,
    ScrollAction.BLOCK, -1, KeyEvent.VK_PAGE_UP,
    InputEvent.CTRL_MASK);
ScrollAction.register(sp, JScrollBar.HORIZONTAL,
    ScrollAction.BLOCK, 1, KeyEvent.VK_PAGE_DOWN,
    InputEvent.CTRL_MASK);
ScrollAction.register(sp, JScrollBar.VERTICAL,
    ScrollAction.BLOCK, -1, KeyEvent.VK_PAGE_UP, 0);
ScrollAction.register(sp, JScrollBar.VERTICAL,
    ScrollAction.BLOCK, 1, KeyEvent.VK_PAGE_DOWN, 0);
    }
}

public class ScrollPaneTest
{
    public static void main(String[] args)
    {
        JFrame frame = new ScrollPaneFrame();
        frame.show();
    }
}

class ScrollAction extends AbstractAction
{
    public ScrollAction(JScrollPane p, int orient,
        int t, int dir)
    {
        scrollPane = p;
        orientation = orient;
        type = t;
        direction = dir;
    }

    public static void register(JScrollPane p, int orient,
        int t, int dir, int key, int modifier)
    {
        p.registerKeyboardAction(
            new ScrollAction(p, orient, t, dir),
            KeyStroke.getKeyStroke(key, modifier, false),
            JComponent.WHEN_IN_FOCUSED_WINDOW);
    }

    public void actionPerformed(ActionEvent evt)
    {
        JScrollBar scrollBar;
        if (orientation == JScrollBar.HORIZONTAL)

```

```

        scrollBar = scrollPane.getHorizontalScrollBar();
    else
        scrollBar = scrollPane.getVerticalScrollBar();
    if (scrollBar == null || !scrollBar.isVisible())
        return;
    int increment;
    if (type == UNIT)
        increment = scrollBar.getUnitIncrement();
    else
        increment = scrollBar.getBlockIncrement();
    scrollBar.setValue(scrollBar.getValue() +
        increment * direction);
}

private JScrollPane scrollPane;
private int orientation; // HORIZONTAL or VERTICAL
private int type; // UNIT or BLOCK
private int direction; // +1 or -1

public static final int UNIT = 1;
public static final int BLOCK = 2;
}

class RulerPanel extends JPanel implements SwingConstants
{
    public RulerPanel(int dir, int w, int h,
        int lblDist, int lbl, int subs)
    {
        direction = dir;
        labelDistance = lblDist;
        label = lbl;
        subdivisions = subs;
        setPreferredSize(new Dimension(w, h));
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        Dimension d = getPreferredSize();
        if (direction == HORIZONTAL)
        {
            int i = 0;
            int x = 0;
            if (subdivisions > 0)
            {
                while (x < d.width)
                {
                    g.drawLine(x, 0, x, (d.height * 4) / 10);
                    i++;
                    x = (i * labelDistance) / subdivisions;
                }
            }
            i = 0;
            x = 0;
            while (x <= d.width)
            {
                g.drawLine(x, 0, x, (d.height * 8) / 10);
                g.drawString("" + i * label, x + 2,
                    (d.height * 8) / 10);
                i++;
                x = i * labelDistance;
            }
        }
    }
}

```

```

    }
}
else
{
    int i = 0;
    int y = 0;
    if (subdivisions > 0)
    {
        while (y <= d.height)
        {
            g.drawLine(0, y, (d.width * 4) / 10, y);
            i++;
            y = (i * labelDistance) / subdivisions;
        }
    }
    i = 0;
    y = 0;
    while (y <= d.height)
    {
        g.drawLine(0, y, (d.width * 8) / 10, y);
        g.drawString("" + i * label, 2, y);
        i++;
        y = i * labelDistance;
    }
}
}

private int direction;
private int labelDistance;
private int subdivisions;
private int label;
}

```

API)

javax.swing.JScrollPane

● JScrollPane(Component c)
가 c

ScrollPane(Component c , int horiz, int vert)

c
horiz HORIZON_SCROLLBAR_ALWAYS, HORIZON_SCROLLBAR_AS_NEEDED,
HORIZON_SCROLLBAR_AS_NEVER
vert VERTICAL_SCROLLBAR_ALWAYS, VERTICAL_SCROLLBAR_AS_NEEDED,
VERTICAL_SCROLLBAR_AS_NEVER

void setRowHeaderView(Component c)

void setColumnHeaderView(Component c)
c 가 .

void setCorner(String where, Component c)

:
where LOWER_LEFT_CORNER, LOWERE_RIGHT_CORNER, UPPER_LEFT_CORNER,
UPPER_RIGHT_CORNER

```

        JScrollPane
        .
        .
        "visible amount"
        JScrollPane
        .
        가 600 400
        , 가 300 200
        가
        200 100 , (200,
        100) (-200, -100)
        600 x 400
        가

```

9-25 :

-
- 가

```

public void adjustmentValueChanged(AdjustmentEvent evt)
{ panel.translate(horiz.getValue(), vert.getValue());
}

```

translate dx, dy ..

```

public void translate(int x, int y)
{ dx = x;
  dy = y;
  repaint();
}

```

```

coordinates) . 0 600 x 0 400 y 가
. paintComponent

```

```

public void paintComponent(Graphics g)
{ super.paintComponent(g);
  g.translate(-dx, -dy);
  g.setColor(Color.red);
  g.drawRect(0, 0, MAX_XWIDTH - 1, MAX_YHEIGHT - 1);
}

```

```

g.setColor(Color.black);
for (int i = 0; i < nsquares; i++)
)

가      가      (0,0)      (600,400)

paintComponent      translate

.

.

.

public boolean mouseDown(Event evt, int x, int y)
{
    x += dx;
    y += dy;
    current = find(x, y);
    . . .
}

가

.      (visible
amount)      가 280*180      (
.      300*200      가
.)      x      599
0      599-290=319      x
가 0      599 가      0      319      . 280

.

가
x y

class ScrollFrame
{
    public ScrollFrame()
    {
        . . .
        addComponentListener(new ComponentAdapter()
        {
            public void componentShown(ComponentEvent evt)
            {
                setVisibleAmounts();
            }
            public void componentResized(ComponentEvent evt)
            {
                setVisibleAmounts();
            }
        });
    }

    public void setVisibleAmounts()
    {
        Dimension d = panel.getSize();
        horiz.setVisibleAmount(d.width);
        vert.setVisibleAmount(d.height);
    }

    가      600*400      가

```

JScrollPane 가

9-14:ScrollTest.java

```

/**
 * @version 1.20 17 Aug 1998
 * @author Cay Horstmann
 */

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class MousePanel extends JPanel
    implements MouseMotionListener
{
    public MousePanel()
    {
        addMouseListener(new MouseAdapter()
        {
            public void mousePressed(MouseEvent evt)
            {
                int x = evt.getX() + dx;
                int y = evt.getY() + dy;
                current = find(x, y);
                if (current < 0) // not inside a square
                {
                    if (x < MAX_XWIDTH && y < MAX_YHEIGHT)
                        add(x, y);
                }
            }

            public void mouseClicked(MouseEvent evt)
            {
                int x = evt.getX() + dx;
                int y = evt.getY() + dy;

                if (evt.getClickCount() >= 2)
                {
                    remove(current);
                }
            }
        });
        addMouseMotionListener(this);
    }

    public void translate(int x, int y)
    {
        dx = x;
        dy = y;
        repaint();
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.translate(-dx, -dy);
        g.setColor(Color.red);
    }
}

```

```

    g.drawRect(0, 0, MAX_XWIDTH - 1, MAX_YHEIGHT - 1);
    g.setColor(Color.black);
    for (int i = 0; i < nsquares; i++)
        draw(g, i);
}

public int find(int x, int y)
{
    for (int i = 0; i < nsquares; i++)
        if (squares[i].x - SQUARELENGTH / 2 <= x &&
            x <= squares[i].x + SQUARELENGTH / 2
            && squares[i].y - SQUARELENGTH / 2 <= y
            && y <= squares[i].y + SQUARELENGTH / 2)
            return i;
    return -1;
}

public void draw(Graphics g, int i)
{
    g.drawRect(squares[i].x - SQUARELENGTH / 2,
        squares[i].y - SQUARELENGTH / 2,
        SQUARELENGTH, SQUARELENGTH);
}

public void add(int x, int y)
{
    if (nsquares < MAXNSQUARES)
    {
        squares[nsquares] = new Point(x, y);
        current = nsquares;
        nsquares++;
        repaint();
    }
}

public void remove(int n)
{
    if (n < 0 || n >= nsquares) return;
    nsquares--;
    squares[n] = squares[nsquares];
    if (current == n) current = -1;
    repaint();
}

public void mouseMoved(MouseEvent evt)
{
    int x = evt.getX();
    int y = evt.getY();

    if (find(x, y) >= 0)
        setCursor(Cursor.getPredefinedCursor(
            Cursor.CROSSHAIR_CURSOR));
    else
        setCursor(Cursor.getDefaultCursor());
}

public void mouseDragged(MouseEvent evt)
{
    int x = evt.getX();
    int y = evt.getY();
}

```

```

        if (current >= 0)
        { Graphics g = getGraphics();
          g.setXORMode(getBackground());
          draw(g, current);
          squares[current].x = x;
          squares[current].y = y;
          draw(g, current);
          g.dispose();
        }
    }

    private static final int SQUARELENGTH = 10;
    private static final int MAXNSQUARES = 100;
    private Point[] squares = new Point[MAXNSQUARES];
    private int nsquares = 0;
    private int current = -1;

    private int dx = 0;
    private int dy = 0;

    public static final int MAX_XWIDTH = 600;
    public static final int MAX_YHEIGHT = 400;
}

class ScrollFrame extends JFrame
    implements AdjustmentListener
{ public ScrollFrame()
    { setTitle("ScrollTest");
      setSize(300, 200);
      addWindowListener(new WindowAdapter()
          { public void windowClosing(WindowEvent e)
              { System.exit(0);
              }
          }
      );

      Container contentPane = getContentPane();
      contentPane.add(panel = new MousePanel(), "Center");
      contentPane.add(vert = new JScrollBar
          (Adjustable.VERTICAL, "East");
      contentPane.add(horiz = new JScrollBar
          (Adjustable.HORIZONTAL, "South");
      vert.addAdjustmentListener(this);
      horiz.addAdjustmentListener(this);
      horiz.setValues(horiz.getValue(), 0, 0,
          MousePanel.MAX_XWIDTH);
      vert.setValues(vert.getValue(), 0, 0,
          MousePanel.MAX_YHEIGHT);

      addComponentListener(new ComponentAdapter()
          { public void componentShown(ComponentEvent evt)
              { setVisibleAmounts();
              }
            public void componentResized(ComponentEvent evt)
              { setVisibleAmounts();
            }
          }
      );
    }
}

```


(native) AWT (motif)

metal look and feel look and feel

가 look and feel 가

가 가

:

가 가 . AWT

Component Container

Component

9-26 :

: 가 JFrame

Component

JApplet

panel.setLayout(new GridLayout(4, 4));

GridLayout

add

panel.add(new JTextField(), "South")

가

가

panel.add(new JCheckBox("italic"));

panel.add(new JCheckBox("bold"));

```

        가 .
        AWT (grid bag
layout) .
        가
        . "grid bag layout"
        .
        .
        (box)
        (sequence)
        가
        " (wrap)"
        2
        가 ,
        (overlay
layout) .
        AWT (card
layout) . (2
        .)
        .
        (grid layout)

```

9-27

9-27:

```

panel.setLayout(new GridLayout(5, 4));

panel.setLayout(new GidLayout(5, 4,3 ,3 ));

(entry)

panel.add(new JButton("1");
panel.add(new JButton("2");

```

9-15

9-15:Calculator.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class CalculatorPanel extends JPanel
    implements ActionListener
{
    public CalculatorPanel()
    {
        setLayout(new BorderLayout());

        display = new JTextField("0");
        display.setEditable(false);
        add(display, "North");

        JPanel p = new JPanel();
        p.setLayout(new GridLayout(4, 4));
        String buttons = "789/456*123-0.=+";
        for (int i = 0; i < buttons.length(); i++)
            add(p, buttons.substring(i, i + 1));
        add(p, "Center");
    }

    private void addButton(Container c, String s)
    {
        JButton b = new JButton(s);
        c.add(b);
        b.addActionListener(this);
    }

    public void actionPerformed(ActionEvent evt)
    {
        String s = evt.getActionCommand();
        if ('0' <= s.charAt(0) && s.charAt(0) <= '9'
            || s.equals("."))
        {
            if (start) display.setText(s);
            else display.setText(display.getText() + s);
            start = false;
        }
        else
        {
            if (start)
            {
                if (s.equals("-"))
                {
                    display.setText(s); start = false; }
                else op = s;
            }
            else
            {
                double x =
                    Double.parseDouble(display.getText());
                calculate(x);
                op = s;
                start = true;
            }
        }
    }
}
```

```

    }

    public void calculate(double n)
    { if (op.equals("+")) arg += n;
      else if (op.equals("-")) arg -= n;
      else if (op.equals("*")) arg *= n;
      else if (op.equals("/")) arg /= n;
      else if (op.equals("=")) arg = n;
      display.setText("" + arg);
    }

    private JTextField display;
    private double arg = 0;
    private String op = "=";
    private boolean start = true;
}

class CalculatorFrame extends JFrame
{ public CalculatorFrame()
  { setTitle("Calculator");
    setSize(200, 200);
    addWindowListener(new WindowAdapter()
      { public void windowClosing(WindowEvent e)
        { System.exit(0);
        }
      } );

    Container contentPane = getContentPane();
    contentPane.add(new CalculatorPanel());
  }
}

public class Calculator
{ public static void main(String[] args)
  { JFrame frame = new CalculatorFrame();
    frame.show();
  }
}

```

가 .
 ()가 .
 . ((gap)
 .) 가

API)
 java.awt.GridLayout

- GridLayout(int rows, int cols)
 GridLayout
 :
 rows
 columns

● GridLayout(int rows, int cols, int hgap, int vgap)
가 GridLayout .

```

:
rows
columns
hgap ( - (overlap)
.)
vgap ( - (overlap)
.)

```

(box layout)

```

. (JPanel . 가 BorderLayout (Box )
FlowLayout .) JPanel
. Box
. Box
가

```

```

.
Box b = Box.createHorizonBox( );

```

```

Box b = Box.createHorizonBox( );

```

```

b.add(okButton)
b.add(cancelButton)

```

```

. . 가
.

```

```

1.가
2.
3. 가 y-alignment
가 getAlignmentY (query) 0 1
. Component 0.5( )

```

```

4. 가 가
5. (preferred)

```

```

. (preferred)
. 가

```

```

. 9-28 9-29 , (box)
. 9-16

```

가 (toggle) .
 (Integer.MAX_VALUE) 가 .

9-28:

(filler)

(filler) (notion) 가 . 3 가 .

- (struts)
- (rigid area)
- (glue)

5 가 .

```
b.add(okButton);
b.add(Box.createHorizontalStrut(5));
b.add(cancelButton);
```

```
b.add(Box.createRigidArea(new Dimension)(5, 20));
```

5 20 가 . 20
 5

가 . - 가 . 가
 . (spring) . (.)

2 가 가 .

```
b.add(okButton);
b.add(Box.createGlue());
b.add(cancelButton);
```

가 okButton
 cancelButton . 가
 9-30

ok 가 .

9-16:BoxLayoutTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class BoxLayoutTest extends JFrame
    implements ActionListener
{
    public BoxLayoutTest()
    {
        setTitle("BoxLayoutTest");
        setSize(300, 200);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        horizontalBox = createBox(true, false);
        verticalBox = createBox(false, false);
        horizontalStrutsAndGlueBox = createBox(true, true);
        verticalStrutsAndGlueBox = createBox(false, true);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(3, 1));
        ButtonGroup directionGroup = new ButtonGroup();

        horizontalButton = addRadioButton(panel,
            directionGroup, "Horizontal", true);
        verticalButton = addRadioButton(panel,
            directionGroup, "Vertical", false);
        strutsAndGlueCheckBox = addCheckBox(panel,
            "Struts and Glue");

        Container contentPane = getContentPane();
        contentPane.add(panel, "South");
        contentPane.add(horizontalBox, "Center");
        currentBox = horizontalBox;
    }

    public Box createBox(boolean horizontal,
        boolean strutsAndGlue)
    {
        Box b;
        if (horizontal)
            b = Box.createHorizontalBox();
        else
            b = Box.createVerticalBox();

        b.add(new JLabel("Name"));
        b.add(new JTextField());

        if (strutsAndGlue)

```

```

        if (horizontal)
            b.add(Box.createHorizontalStrut(5));
        else
            b.add(Box.createVerticalStrut(5));

        b.add(new JLabel("Password"));
        b.add(new JTextField());

        if (strutsAndGlue)
            b.add(Box.createGlue());

        b.add(new JButton("Ok"));

        return b;
    }

    public JRadioButton addRadioButton(JPanel p,
        ButtonGroup g, String name, boolean selected)
    { JRadioButton button
        = new JRadioButton(name, selected);
        button.addActionListener(this);
        g.add(button);
        p.add(button);
        return button;
    }

    public JCheckBox addCheckBox(JPanel p, String name)
    { JCheckBox checkBox = new JCheckBox(name);
        checkBox.addActionListener(this);
        p.add(checkBox);
        return checkBox;
    }

    public void actionPerformed(ActionEvent evt)
    { Container contentPane = getContentPane();
        contentPane.remove(currentBox);

        if (horizontalButton.isSelected())
        { if (strutsAndGlueCheckBox.isSelected())
            currentBox = horizontalStrutsAndGlueBox;
            else
            currentBox = horizontalBox;
        }
        else
        { if (strutsAndGlueCheckBox.isSelected())
            currentBox = verticalStrutsAndGlueBox;
            else
            currentBox = verticalBox;
        }

        contentPane.add(currentBox, "Center");
        contentPane.validate(); // force layout
        repaint();
    }
}

```

```

public static void main(String[] args)
{
    Frame f = new BoxLayoutTest();
    f.show();
}

private Box horizontalBox;
private Box verticalBox;
private Box horizontalStrutsAndGlueBox;
private Box verticalStrutsAndGlueBox;
private Box currentBox;

private JCheckBox strutsAndGlueCheckBox;
private JRadioButton horizontalButton;
private JRadioButton verticalButton;
}

```

API)

javax.swing.Box

- static Box createHorizontalBox()
- static Box createVerticalBox()

- static Component createHorizontalGlue()
- static Component createVerticalGlue()
- static Component createGlue()

- static Component createHorizontalStrut(int width)
- static Component createVerticalStrut(int height)
- static Component createRigidArea(Dimension d)

API)

java.awt.Component

- float getAlignmentX()
- float getAlignmentY()

x	y	0	1.0	(alignment)	. 0
0.5		1		.	

(Grid Bag Layout)

가
가 (HTML
)

9-31

-
-
-
-
-

9-31 :

9-31 4 3 3 2 3

9-32 :

1. GridBagLayout . (가 가 가
 2. GridBagLayout 가 가
 3. GridBagConstraints . (GridBagConstraints 가 .)
 4. , GridBagConstraints setConstraints GridBagLayout .
- 가 .
- . (
- .)

```
GridBagLayout layout = new GridBagLayout();
panel.setLayout(layout);
GridBagConstraints constraints = new GridBagConstraints();
constraints.gridx = 0;
constraints.gridy = 0;
constraints.gridwidth = 1;
constraints.gridheight = 3;
JList style = new JList(4);
layout.setConstraints(style, constraints);
panel.add(style);
```

(.)

```

가 GridBagConstraints
가
gridx, gridy, gridwidth, gridheight
. gridx gridy 가
. gridwidth gridheight 가
.
weight
weight (weightx, weighty)
. weight 0 ,
가 .
weighty 0 가 weight 0
, 가
weight 가 " "
. 100 가
weight 0 , weight
.
fill, anchor
. 가 가 fill
GridBagConstraints.NONE, GridBagConstraints.HORIZONTAL,
GridBagConstraints.VERTICAL, GridBagConstraints.BOTH
가 , anchor
GridBagConstraints.NORTH,
GridBagConstraints.NORTHEAST, GridBagConstraints.EAST, ...
(padding)
가 GridBagLayout insert
. Inserts left, top, right, bottom
. (external)
ipadx ipady (internal padding)
. 가
.
gridx, gridy, gridwidth, gridheight
AWT gridx, gridy
GridBagConstraints.RELATIVE
( ,

```

```

        )      가      .

        gridwidth, gridheight      .
가
GridBagConstraints.REMAINDER      .
가      .      가
        RELATIVE      .

        .

        .

-----
Grid Bag Layout
1.
2.      가

3.      0, 1, 2, 3      gridx, gridy,
   gridwidth  gridheight      .
4.      가?      fill  anchor      가      .
5.      weight  100      weightx  weighty  0
        .
6.      . GridBagConstraints      .
   (constraint)가
7.
-----

```

9-17

9-17: FontDialog.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class FontDialog extends JFrame
    implements ActionListener, ListSelectionListener
{
    public FontDialog()
    {
        setTitle("FontDialog");
        setSize(300, 200);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        Container contentPane = getContentPane();
        GridBagLayout gbl = new GridBagLayout();
        contentPane.setLayout(gbl);

        style = new JList(new String[]

```

```

        { "Serif", "SansSerif", "Monospaced",
          "Dialog", "DialogInput"
        });
style.setSelectedIndex(0);

bold = new JCheckBox("Bold");
italic = new JCheckBox("Italic");
JLabel label = new JLabel("Size: ");
size = new JTextField("10", 2);
sample = new JTextField();
sample.setEditable(false);

GridBagConstraints gbc = new GridBagConstraints();
gbc.fill = GridBagConstraints.BOTH;
gbc.weightx = 0;
gbc.weighty = 100;
add(style, gbc, 0, 0, 1, 3);
gbc.weightx = 100;
gbc.fill = GridBagConstraints.NONE;
gbc.anchor = GridBagConstraints.CENTER;
add(bold, gbc, 1, 0, 2, 1);
add(italic, gbc, 1, 1, 2, 1);
add(label, gbc, 1, 2, 1, 1);
gbc.fill = GridBagConstraints.HORIZONTAL;
add(size, gbc, 2, 2, 1, 1);
gbc.anchor = GridBagConstraints.SOUTH;
gbc.weighty = 0;
add(sample, gbc, 0, 3, 4, 1);
sample.setText("The quick brown fox");

bold.addActionListener(this);
italic.addActionListener(this);
style.addListSelectionListener(this);
size.addActionListener(this);
}

public void add(Component c, GridBagConstraints gbc,
    int x, int y, int w, int h)
{
    gbc.gridx = x;
    gbc.gridy = y;
    gbc.gridwidth = w;
    gbc.gridheight = h;
    getContentPane().add(c, gbc);
}

public void valueChanged(ListSelectionEvent evt)
{
    updateFont();
}

public void actionPerformed(ActionEvent evt)
{
    updateFont();
}

public void updateFont()

```

```

{ Font font =
    new Font((String)style.getSelectedValue(),
        (bold.isSelected() ? Font.BOLD : 0)
        + (italic.isSelected() ? Font.ITALIC : 0),
        Integer.parseInt(size.getText()));
    sample.setFont(font);
    repaint();
}

```

```

public static void main(String[] args)
{ Frame f = new FontDialog();
  f.show();
}

```

```

private JList style;
private JCheckBox bold;
private JCheckBox italic;
private JTextField size;
private JTextField sample;
}

```

API)

java.awt.GridBagConstraints

- int gridx, gridy
가 .
- int gridwidth, gridheight
가 .
- double weightx, weighty
가 .
- int anchor
가 . (CENTER, NORTH, NORTHEAST, EAST, SOUTHEAST, SOUTH, SOUTHWEST, WEST, NORTHWEST)
- int fill
가 . (NONE, BOTH, HORIZONTAL, VERTICAL)
- int ipadx, ipady
" " 가 .
- Insets insets
" " 가 .
- GridBagConstraints(int gridx, int gridy, int gridwidth, int gridheight, double weightx, double weighty, int anchor, int fill, Insets insets, int ipadx, int ipady)
가 GridBagConstraints . (Sun)

가

가 .

-
-
-

가 .

```
panel.setLayout(null);
Button ok = new Button("Ok");
ok.reshape(10, 10, 30, 15);
```

API)

java.awt.Component

- void reshape(int x, int y, int width, int height)

```
:
x, y
width, height
```

LayoutManager
7-19

9-33 :

LayoutManager (implement)

```
void addLayoutComponent(String s, Component c);
void removeLayoutComponent(Component c);
Dimension preferredLayoutSize(Container parent);
Dimension minimumLayoutSize(Container parent);
void layoutContainer(Container parent);
```

가 가

가

가

reshape

```
-----
: AWT   LayoutManger2      5      10      가
      가      . LayoutManger2
가      add      .      BorderLayout
GridBagLayout   LayoutManger2      .
-----
```

9-18 CircleLayout

9-18:CircleLayoutTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class CircleLayoutFrame extends JFrame
{ public CircleLayoutFrame()
  { setTitle("CircleLayoutTest");
    setSize(300, 300);
    addWindowListener(new WindowAdapter()
      { public void windowClosing(WindowEvent e)
        { System.exit(0);
        }
      }
    );

    getContentPane().setLayout(new CircleLayout());
    getContentPane().add(new Button("Yes"));
    getContentPane().add(new Button("No"));
    getContentPane().add(new Button("Ok"));
    getContentPane().add(new Button("Cancel"));
    getContentPane().add(new Button("Abort"));
    getContentPane().add(new Button("Retry"));
    getContentPane().add(new Button("Ignore"));
  }
}

class CircleLayout implements LayoutManager
{ public void addLayoutComponent(String name,
  Component comp)
  {}

  public void removeLayoutComponent(Component comp)
  {}

  public void setSizes(Container parent)
  { if (sizesSet) return;
    int n = parent.getComponentCount();

    preferredWidth = 0;
    preferredHeight = 0;
    minWidth = 0;
    minHeight = 0;
    maxComponentWidth = 0;
    maxComponentHeight = 0;

    for (int i = 0; i < n; i++)
```

```

    { Component c = parent.getComponent(i);
      if (c.isVisible()) {
        Dimension d = c.getPreferredSize();
        maxComponentWidth = Math.max(maxComponentWidth,
            d.width);
        maxComponentHeight = Math.max(maxComponentWidth,
            d.height);
        preferredHeight += d.height;
      }
    }
    preferredHeight += maxComponentHeight;
    preferredWidth = 2 * maxComponentWidth;
    minHeight = preferredHeight;
    minWidth = preferredWidth;
    sizesSet = true;
}

public Dimension preferredLayoutSize(Container parent)
{ Dimension dim = new Dimension(0, 0);
  setSizes(parent);
  Insets insets = parent.getInsets();
  dim.width = preferredWidth + insets.left
      + insets.right;
  dim.height = preferredHeight + insets.top
      + insets.bottom;
  return dim;
}

public Dimension minimumLayoutSize(Container parent)
{ Dimension dim = new Dimension(0, 0);
  setSizes(parent);
  Insets insets = parent.getInsets();
  dim.width = minWidth + insets.left + insets.right;
  dim.height = minHeight + insets.top + insets.bottom;
  return dim;
}

public void layoutContainer(Container parent)
{ Insets insets = parent.getInsets();
  int containerWidth = parent.getSize().width
      - insets.left - insets.right;
  int containerHeight = parent.getSize().height
      - insets.top - insets.bottom;
  int xradius = (containerWidth - maxComponentWidth)
      / 2;
  int yradius = (containerHeight - maxComponentHeight)
      / 2;

  setSizes(parent);
  int xcenter = insets.left + containerWidth / 2;
  int ycenter = insets.top + containerHeight / 2;

  int n = parent.getComponentCount();
  for (int i = 0; i < n; i++)

```

```

        { Component c = parent.getComponent(i);
          if (c.isVisible())
          { Dimension d = c.getPreferredSize();
            double angle = 2 * Math.PI * i / n;
            int x = xcenter
              + (int)(Math.cos(angle) * xradius);
            int y = ycenter
              + (int)(Math.sin(angle) * yradius);

            c.setBounds(x - d.width / 2, y - d.height / 2,
              d.width, d.height);
          }
        }
      }

      private int minWidth = 0;
      private int minHeight = 0;
      private int preferredWidth = 0, preferredHeight = 0;
      private boolean sizesSet = false;
      private int maxComponentWidth = 0;
      private int maxComponentHeight = 0;
    }

    public class CircleLayoutTest {
      public static void main(String[] args)
      { JFrame f = new CircleLayoutFrame();
        f.show();
      }
    }

```

9-18 CircleLayout 가

API)

java.awt.LayoutManager

- void addLayoutComponent(String name, Component comp)
가 .

:
name
comp 가

- void removeLayoutComponent(Component c)
.

:
comp 가

- Dimension preferredLayoutSize(Container parent)
.

:
parent 가

- Dimension minimumLayoutSize(Container parent)
.

`style.setNextFocusableComponent(size);`

- 1.
- 2.
- 3.
- 4.
- 5.

`setNextFocusableComponent` 가

API)

`java.awt.Component`

- `void requestFocus( )`
가

API)

`java.swing.JComponent`

- `void setNextFocusableComponent(Component c)`
c

Window Motif (menu bar) (submenu) item) 가 9-35 가 (menu

9-35: 가

:	<code>(toolbar)</code>
<code>JToolBar</code>	가 2

가

`JMenuBar menubar = new JMenuBar();`

가 setJMenuBar

`frame.setJMenuBar(menuBar);`

```

        ,
        .

JMenu editMenu = new JMenu("Edit");

        ,
        가 .

JMenuItem pasteItem = new JMenuItem("Paste");
editMenu.add(pasteItem);
editMenu.addSeparator();
JMenu optionsMenu = . . . ; // a submenu
editMenu.add(optionsMenu);
. . .

        가 .

menubar.addMenu(editMenu);

        가
        ,
        가 .

pasteItem.addActionListener(this);

        가
        .

JMenu menu = new JMenu("Edit");
Item = new JMenuItem("Cut");
Item.addActionListener(this);
menu.add(item);
Item = new JMenuItem("Copy");
Item.addActionListener(this);
menu.add(item);
Item = new JMenuItem("Paste");
Item.addActionListener(this);
menu.add(item);
menuBar.add(menu);

        makeMenu
        .
        3
        , makeMenu
        ,
        null
        . makeMenu
        null
        가
        . (
        가
        makeMenu
        .

menuBar.add(makeMenu("Edit", new Object[]
{
    "Cut", "Copy", "Paste", null,
    makeMenu(" Options", new Object[]
    {
        "Insert", "Overtyp", "Read only"
    }, this)
}, this));

```

가 makeMenu

가 .

```
Public static JMenu makeMenu (Object parent, Object[] items,
Object target)
{
    JMenu m = null;
    if (parent instanceof JMenu)
        m = (JMenu)parent;
    else if (parent instanceof String)
        m = new JMenu((String)parent);
    else
        return null;

    for (int i=0; i<item.length; i++)
    {
        if (item[i] == null)
            m.addSeparator();
        else
            m.add(makeMenuItem(items[i], target));
    }

    return m;
}

public static JMenuItem makeMenuItem (Object item, Object target)
{
    JMenuItem r = null;
    if (item instanceof String)
        r = new JMenuItem((String)item);
    else if (item instanceof JMenuItem)
        r = (JMenuItem) item;
    else return null;

    if (target instanceof ActionListener)
        r.addActionListener((ActionListener)target);
    return r;
}
```

: , VB

Windows 가 .

: 가 JMenu.add(String s)

가 . , editMenu.add("Paste");

add

가 .

JMenuItem pasteItem = editMenu.add("Paste");
PasteItem.addActionListener(this);

```

        가                actionPerformed
        .                instanceof
        .                getActionCommand

        .

public void actionPerformed(Event evt)
{
    if (evt.getSource() instanceof JMenuItem)
        String arg = evt.getActionCommand();
        If (arg.equals("Open")) . . .
        else if (arg.equals("Save")) . . .
        . . .
    }
}

```

```

: 8      Action                가
        .      AbstractAction      Action
        .      AbstractAction
        .      JMenuItem      add(Action)
        ,      가
        ,

Action openaction = new AbstractAction("Open")
{
    public void actionPerformed(ActionEvent evt)
    {
        // open handler goes here
    }
};
menu.add(openAction);

8

```

```

javax.swing.JMenu
● JMenuItem(String label)
    : label

● JMenuItem add(JMenuItem item)
    가
    : item    가

● JMenuItem add(String label)
    가
    :    label

● JMenuItem add(Action a)
    가
    :    a    ,

● void addSeparator()
    가

● void insert(JMenuItem menu, int index)
    (    )    가

```

```

        : menu 가
        index 가

● void insertSeparator(int index)
    가
    : index 가

● void remove(int index)
    : index

● void remove(JMenuItem item)
    : item

javax.swing.JMenuItem
● JMenuItem(String label)
    : label

javax.swing.JFrame
● void setJMenuBar(JMenuBar member)

, JMenuItem AbstractButton
가 , JMenuItem JMenuItem
JMenuItem(String, Icon) JMenuItem(Icon)
AbstractButton JMenuItem
setIcon

JMenuItem cutItem = new JMenuItem("Cut", new ImageIcon("cut.gif"));

9-36
, JMenuItem
, AbstractButton JMenuItem
setHorizontalTextPosition

cutItem.setHorizontalTextPosition(SwingConstants.RIGHT);

9-36:

javax.swing.JMenuItem
● JMenuItem(String label, Icon icon)
    : label
    icon

javax.swing.AbstractButton
● void setHorizontalTextPosition(int pos)

```

```

        : pos SwingConstants.RIGHT(
            ), SwingConstants.LEFT
            SwingConstants.CENTER

        (check box) (radio button)
        .( 9-37 ) 가 ,
        .

        ,

        JCheckBoxMenuItem readonlyItem = new JCheckBoxMenuItem("Ready-only");
        OptionMenu.add(readonlyItem);

        가 . ,
        .

        ButtonGroup group = new ButtonGroup();
        JRadioButtonMenuItem insertItem = new JRadioButtonMenuItem("Insert");
        InsertItem.setSelected(true);
        JRadioButtonMenuItem overtypeItem = new JRadioButtonMenuItem
        ("Overtyping");
        group.add(insertItem);
        group.add(overtypingItem);
        optionMenu.add(insertItem);
        optionMenu.add(overtypingItem);

        , 가
        . isSelected
        .( , '
        .) setSelected
        .

```

9-37:

javax.swing.JCheckBoxMenuItem

- JCheckBoxMenuItem(String label)
- JCheckBoxMenuItem(String label, boolean state) .(true
.)

javax.swing.JRadioButtonMenuItem

- JRadioButtonMenuItem(String label)
- JRadioButtonMenuItem(String label, boolean state) .(true
.)

javax.swing.AbstractButton

- `void setSelected(boolean state)`

Pop-up

9-38:

```
JPopupMenu popup = new JPopupMenu();
```

```
JMenuItem item = new JMenuItem("Cut");
Item.addActionListener(this);
popup.add(item);
```

```
popup.show(panel, x, y);
```

- .
- 가 .

가 .

:	JDK	isPopupTrigger	mouseReleased
		mousePressed	mouseClicked

javax.swing.JpopupMenu

$\begin{matrix} : & c & & 가 \\ & x, y & & c & & 가 & - \end{matrix}$

java.awt.event.MouseEvent

● boolean isPopupTrigger()

가 true .

(keyboard mnemonics)

. ,
.

JMenuItem cutItem = new JMenuItem("Cut", "T");

.(9-39)

가 , 가 , 가 .(
가 .)

가 가 setMnemonic . ,

JMenu helpMenu = new JMenu("Help");
helpMenu.setMnemonic('H');

Help ALT+
ALT+H . ,

9-39:

(accelerators)

, CTRL+O, CTRL+S .
setAccelerator .

SetAccelerator Keystroke . ,
openItem CTRL+O .

OpenItem.setAccelerator(Keystroke.getKeyStroke(KeyEvent.VK_O,
InputEvent.CTRL_MASK));

가 가

.(9-40)

9-40:


```

public void menuSelected(MenuEvent evt)
{
    saveItem.setEnabled(!readonlyItem.isSelected());
    saveAsItem.setEnabled(!readonlyItem.isSelected());
}

```

```

menuDeselected          가          ,          . MenuCanceled
                           가
MenuAdapter              가          2

```

```

javax.swing.JMenuItem
● void setEnabled(boolean b)

```

```

javax.swing.event.MenuListener
● void menuSelected(MenuEvent e)
    가
● void menuDeselected(MenuEvent e)
    가

```

```

● void menuCanceled(MenuEvent e)
    ,          가

```

9-19

9-19: MenuTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class MenuTest extends JFrame
    implements ActionListener, MenuListener
{
    public MenuTest()
    {
        setTitle("MenuTest");
        setSize(400, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        JMenuBar mbar = new JMenuBar();
        setJMenuBar(mbar);

        // demonstrate enabled/disabled items

        JMenu fileMenu = new JMenu("File");
        fileMenu.addMenuListener(this);
    }
}

```

```

// demonstrate accelerators

JMenuItem openItem = new JMenuItem("Open");
openItem.setAccelerator
    (KeyStroke.getKeyStroke(KeyEvent.VK_O,
        InputEvent.CTRL_MASK));
saveItem = new JMenuItem("Save");
saveItem.setAccelerator
    (KeyStroke.getKeyStroke(KeyEvent.VK_S,
        InputEvent.CTRL_MASK));
saveAsItem = new JMenuItem("Save As");

mbar.add(makeMenu(fileMenu,
    new Object[]
    { "New",
      openItem,
      null,
      saveItem,
      saveAsItem,
      null,
      "Exit"
    },
    this));

// demonstrate check box and radio button menus

readonlyItem = new JCheckBoxMenuItem("Read-only");
ButtonGroup group = new ButtonGroup();
JRadioButtonMenuItem insertItem
    = new JRadioButtonMenuItem("Insert");
insertItem.setSelected(true);
JRadioButtonMenuItem overtypeItem
    = new JRadioButtonMenuItem("Overtype");
group.add(insertItem);
group.add(overtimeItem);

// demonstrate icons and nested menus

mbar.add(makeMenu("Edit",
    new Object[]
    { new JMenuItem("Cut",
        new ImageIcon("cut.gif")),
      new JMenuItem("Copy",
        new ImageIcon("copy.gif")),
      new JMenuItem("Paste",
        new ImageIcon("paste.gif")),
      null,
      makeMenu("Options",
          new Object[]
          { readonlyItem,
            null,
            insertItem,
            overtypeItem
          }
      )
    },
    this));

```

```

        },
        this)
    },
    this));

// demonstrate mnemonics

JMenu helpMenu = new JMenu("Help");
helpMenu.setMnemonic('H');

mbar.add(makeMenu(helpMenu,
    new Object[]
    { new JMenuItem("Index", 'I'),
      new JMenuItem("About", 'A')
    },
    this));

// demonstrate pop-ups

popup = makePopupMenu(
    new Object[]
    { "Cut",
      "Copy",
      "Paste"
    },
    this);

getContentPane().addMouseListener(new MouseAdapter()
{
    public void mouseReleased(MouseEvent evt)
    {
        if (evt.isPopupTrigger())
            popup.show(evt.getComponent(),
                evt.getX(), evt.getY());
    }
});
}

public void actionPerformed(ActionEvent evt)
{
    String arg = evt.getActionCommand();
    System.out.println(arg);
    if(arg.equals("Exit"))
        System.exit(0);
}

public void menuSelected(MenuEvent evt)
{
    saveItem.setEnabled(!readonlyItem.isSelected());
    saveAsItem.setEnabled(!readonlyItem.isSelected());
}

public void menuDeselected(MenuEvent evt)
{
}

public void menuCanceled(MenuEvent evt)
{
}

```

```

}

public static JMenu makeMenu(Object parent,
    Object[] items, Object target)
{
    JMenu m = null;
    if (parent instanceof JMenu)
        m = (JMenu)parent;
    else if (parent instanceof String)
        m = new JMenu((String)parent);
    else
        return null;

    for (int i = 0; i < items.length; i++)
    {
        if (items[i] == null)
            m.addSeparator();
        else
            m.add(makeMenuItem(items[i], target));
    }

    return m;
}

public static JMenuItem makeMenuItem(Object item,
    Object target)
{
    JMenuItem r = null;
    if (item instanceof String)
        r = new JMenuItem((String)item);
    else if (item instanceof JMenuItem)
        r = (JMenuItem)item;
    else return null;

    if (target instanceof ActionListener)
        r.addActionListener((ActionListener)target);
    return r;
}

public static JPopupMenu makePopupMenu
    (Object[] items, Object target)
{
    JPopupMenu m = new JPopupMenu();

    for (int i = 0; i < items.length; i++)
    {
        if (items[i] == null)
            m.addSeparator();
        else
            m.add(makeMenuItem(items[i], target));
    }

    return m;
}

public static void main(String[] args)
{
    Frame f = new MenuTest();
    f.show();
}

```

[illegible]

●

-
-

가 가 . 가
가

.

look feel .

9-42:

(*message type*) .5 가

.

ERROR_MESSAGE
INFORMATION_MESSAGE
WARNING_MESSAGE
QUESTION_MESSAGE
PLAIN_MESSAGE

PLAIN_MESSAGE

.

.

, . 가 , , .

String: .
Icon: .
Component: .
Object[]: ,
. toString
.

9-19

. paintComponent
Component

. 가

.

(*option type*) .

showMessageDialog showInputDialog (Ok,
Ok/Cancel) . showConfirmDialog 4 가

.

DEFAULT_OPTION
YES_NO_OPTION
YES_NO_CANCEL_OPTION
OK_CANCEL_OPTION

```

        showOptionDialog
        .

String:
Icon:
Component:
        :
        toString

        .

showMessageDialog
showConfirmDialog
showOptionDialog
showInputDialog
        가

showConfirmDialog    showOptionDialog    가    가
        .
        가    ' CLOSED_OPTION
        ,

OK_OPTION
CANCEL_OPTION
YES_OPTION
NO_OPTION
CLOSED_OPTION

1.
2.
3.
4.
        Ok/Cancel)
5.
6.
7. JOptionPane API
        ,    9-42    가
        .
        OK_CANCEL_
OPTION
        .

int selection = JOptionPane.showConfirmDialog(parent, "Message",
        "Title", JOptionPane.OK_CANCEL_OPTION,
        JOptionPane.WARNING_MESSAGE);
If (selection == JOptionPane.OK_OPTION)


---


        :    ('\n')


---



```

9-43: OptionDialogTest

9-20

9-43

9-20: OptionDialogTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;

class ButtonPanel extends JPanel
{
    public ButtonPanel(String title, String[] options)
    {
        setBorder(BorderFactory.createTitledBorder
            (BorderFactory.createEtchedBorder(), title));
        setLayout(new BoxLayout(this,
            BoxLayout.Y_AXIS));
        group = new ButtonGroup();

        for (int i = 0; i < options.length; i++)
        {
            JRadioButton b = new JRadioButton(options[i]);
            b.setActionCommand(options[i]);
            add(b);
            group.add(b);
            b.setSelected(i == 0);
        }
    }

    String getSelection()
    {
        return group.getSelection().getActionCommand();
    }

    ButtonGroup group;
}

public class OptionDialogTest extends JFrame
    implements ActionListener
{
    public OptionDialogTest()
    {
        setTitle("OptionDialogTest");
        setSize(600, 400);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }

    JPanel gridPanel = new JPanel();
    gridPanel.setLayout(new GridLayout(2, 3));

    typePanel = new ButtonPanel("Type",
        new String[]
        {
            "Message",
            "Confirm",
        }
    );
}
```

```

        "Option",
        "Input"
    });

    messageTypePanel = new ButtonPanel("Message Type",
        new String[]
        { "ERROR_MESSAGE",
          "INFORMATION_MESSAGE",
          "WARNING_MESSAGE",
          "QUESTION_MESSAGE",
          "PLAIN_MESSAGE"
        });

    messagePanel = new ButtonPanel("Message",
        new String[]
        { "String",
          "Icon",
          "Component",
          "Other",
          "Object[]"
        });

    optionTypePanel = new ButtonPanel("Confirm",
        new String[]
        { "DEFAULT_OPTION",
          "YES_NO_OPTION",
          "YES_NO_CANCEL_OPTION",
          "OK_CANCEL_OPTION"
        });

    optionsPanel = new ButtonPanel("Option",
        new String[]
        { "String[]",
          "Icon[]",
          "Object[]"
        });

    inputPanel = new ButtonPanel("Input",
        new String[]
        { "Text field",
          "Combo box"
        });

    JPanel showPanel = new JPanel();
    JButton showButton = new JButton("Show");
    showButton.addActionListener(this);
    showPanel.add(showButton);

    gridPanel.add(typePanel);
    gridPanel.add(messageTypePanel);
    gridPanel.add(messagePanel);
    gridPanel.add(optionTypePanel);
    gridPanel.add(optionsPanel);
    gridPanel.add(showPanel);

```

```

gridPanel.add(inputPanel);

    Container contentPane = getContentPane();
    contentPane.add(gridPanel, "Center");
    contentPane.add(showPanel, "South");
}

public Object getMessage()
{
    String s = messagePanel.getSelection();
    if (s.equals("String"))
        return messageString;
    else if (s.equals("Icon"))
        return messageIcon;
    else if (s.equals("Component"))
        return messageComponent;
    else if (s.equals("Object[]"))
        return new Object[]
        {
            messageString,
            messageIcon,
            messageComponent,
            messageFont
        };
    else if (s.equals("Other"))
        return messageFont;
    else return null;
}

public Object[] getOptions()
{
    String s = optionsPanel.getSelection();
    if (s.equals("String[]"))
        return new String[] { "Yellow", "Blue", "Red" };
    else if (s.equals("Icon[]"))
        return new Icon[]
        {
            new ImageIcon("yellow-ball.gif"),
            new ImageIcon("blue-ball.gif"),
            new ImageIcon("red-ball.gif")
        };
    else if (s.equals("Object[]"))
        return new Object[]
        {
            messageString,
            messageIcon,
            messageComponent,
            messageFont
        };
    else
        return null;
}

public int getType(ButtonPanel panel)
{
    String s = panel.getSelection();
    try
    {
        Class cl = JOptionPane.class;
        return cl.getField(s).getInt(cl);
    }
}

```

```

        catch(Exception e)
        { return -1;
        }
    }

    public void actionPerformed(ActionEvent evt)
    { if (typePanel.getSelection().equals("Confirm"))
        JOptionPane.showConfirmDialog(this,
            getMessage(),
            "Title",
            getType(optionTypePanel),
            getType(messageTypePanel));
        else if (typePanel.getSelection().equals("Input"))
        { if (inputPanel.getSelection().equals("Text field"))
            JOptionPane.showInputDialog(this,
                getMessage(),
                "Title",
                getType(messageTypePanel));
            else
            JOptionPane.showInputDialog(this,
                getMessage(),
                "Title",
                getType(messageTypePanel),
                null,
                new String[] { "Yellow", "Blue", "Red" },
                "Blue");
        }
        else if (typePanel.getSelection().equals("Message"))
            JOptionPane.showMessageDialog(this,
                getMessage(),
                "Title",
                getType(messageTypePanel));
        else if (typePanel.getSelection().equals("Option"))
            JOptionPane.showOptionDialog(this,
                getMessage(),
                "Title",
                getType(optionTypePanel),
                getType(messageTypePanel),
                null,
                getOptions(),
                getOptions()[0]);
    }

    public static void main(String[] args)
    { JFrame f = new OptionDialogTest();
      f.show();
    }

    private ButtonPanel typePanel;
    private ButtonPanel messagePanel;
    private ButtonPanel messageTypePanel;
    private ButtonPanel optionTypePanel;
    private ButtonPanel optionsPanel;
    private ButtonPanel inputPanel;

```

```

private String messageString = "Message";
private Icon messageIcon
    = new ImageIcon("blue-ball.gif");
private Font messageFont
    = new Font("Serif", Font.PLAIN, 8);
private Component messageComponent
    = new JPanel()
    {
        public void paintComponent(Graphics g)
        {
            super.paintComponent(g);
            g.setFont(messageFont);
            g.drawString("Component", 0, 8);
        }
        public Dimension getMinimumSize()
        {
            return new Dimension(12, 30);
        }
    };
}

```

javax.swing.JOptionPane

- static void showMessageDialog(Component parent, Object message, String title, int messageType, Icon icon)
- static void showMessageDialog(Component parent, Object message, String title, int messageType)
- static void showMessageDialog(Component parent, Object message)
- static void showInternalMessageDialog(Component parent, Object message, String title, int messageType, Icon icon)
- static void showInternalMessageDialog(Component parent, Object message, String title, int messageType)
- static void showInternalMessageDialog(Component parent, Object message)

```

        .(
        .)
:      parent      (null      .)
      message
      title
      messageType      ERROR_MESSAGE,  INFORMATION_MESSAGE,
                        WARNING_MESSAGE, QUESTION_MESSAGE,
                        PLAIN_MESSAGE
      icon

```

- static int showConfirmDialog(Component parent, Object message, String title, int optionType, int messageType, Icon icon)
- static int showConfirmDialog(Component parent, Object message, String title, int optionType, int messageType)
- static int showConfirmDialog(Component parent, Object message,

String title, int optionType)

- static int showConfirmDialog(Component parent, Object message)
- static int showInternalConfirmDialog(Component parent, Object message, String title, int optionType, int messageType, Icon icon)
- static int showInternalConfirmDialog(Component parent, Object message, String title, int optionType, int messageType)
- static int showInternalConfirmDialog(Component parent, Object message, String title, int optionType)
- static int showInternalConfirmDialog(Component parent, Object message)

```

                                .(
                                .) (OK_OPTION, CANCEL_OPTION,
YES_OPTION, NO_OPTION, CLOSED_OPTION ) 가
                                CLOSED_OPTION .
                                : parent (null .)
                                message
                                title
                                messageType ERROR_MESSAGE, INFORMATION_MESSAGE,
                                WARNING_MESSAGE, QUESTION_MESSAGE,
                                PLAIN_MESSAGE
                                OptionType OK_OPTION, CANCEL_OPTION,
                                YES_OPTION, NO_OPTION,
                                CLOSED_OPTION
                                icon

```

- static int showOptionDialog(Component parent, Object message, String title, int optionType, int messageType, Icon icon, Object[] options, Object default)
- static int showInternalOptionDialog(Component parent, Object message, String title, int optionType, int messageType, Icon icon, Object[] options, Object default)

```

                                .(
                                .)
가                                CLOSED_OPTION .
                                : parent (null .)
                                message
                                title
                                messageType ERROR_MESSAGE, INFORMATION_MESSAGE,
                                WARNING_MESSAGE, QUESTION_MESSAGE,
                                PLAIN_MESSAGE
                                OptionType OK_OPTION, CANCEL_OPTION,
                                YES_OPTION, NO_OPTION,
                                CLOSED_OPTION

```

```

        icon
        options                (      ,      ,
                                .)
        default

```

- static Object showInputDialog(Component parent, Object message, String title, int messageType, Icon icon, Object[] values, Object default)
- static String showInputDialog(Component parent, Object message, String title, int messageType)
- static String showInputDialog(Component parent, Object message)
- static String showInputDialog(Object message)
- static Object showInternalInputDialog(Component parent, Object message, String title, int messageType, Icon icon, Object[] values, Object default)
- static String showInternalInputDialog(Component parent, Object message, String title, int messageType)
- static String showInternalInputDialog(Component parent, Object message)

```

                                .(
                                가
        .)
        null
:      parent                (null      .)
        message
        title
        messageType          ERROR_MESSAGE, INFORMATION_MESSAGE,
                                WARNING_MESSAGE, QUESTION_MESSAGE,
                                PLAIN_MESSAGE
        icon
        options                (      ,      ,
                                .)
        default

```

가 . JOptionPane .

9-44 가 , About

1. JFrame , JDialog , JDialog , 가

```

        Boolean
        가
        가
        null
        가
2.
3.
4.

```

9-44: About

```

        가
        .

public AboutDialog(JFrame parent)
{
    super(parent, "About DialogTest", true);

    Box b = Box.createVerticalBox();
    b.add(Box.createGlue());
    b.add(new JLabel("Core Java"));
    b.add(new JLabel("By Cay Horstmann and Gary Cornell"));
    b.add(Box.createGlue());
    getContentPane().add(b, "Center");

    JPanel p2 = new JPanel();
    JButton ok = new JButton("Ok");
    P2.add(ok);
    getContentPane().add(p2, "South");

    ok.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent evt)
        {
            setVisible(false);
        }
    });

    setSize(250, 150);
}

        ,
        가
        ,
        가
        .
        show

        .

JDialog dialog = new AboutDialog(this);
dialog.show();

        ,
        About

        .

if (dialog == null) // first time
    dialog = new AboutDialog(this);
dialog.show();

```

가 Ok , . Ok

```
ok.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent evt)
    {
        setVisible(false); }
} );
```

가 "close" , .
JFrame 가 , setDefaultCloseOperation

9-21 About .

9-21: DialogTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class DialogFrame extends JFrame
    implements ActionListener
{
    public DialogFrame()
    {
        setTitle("DialogTest");
        setSize(300, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        JMenuBar mbar = new JMenuBar();
        setJMenuBar(mbar);
        JMenu fileMenu = new JMenu("File");
        mbar.add(fileMenu);
        JMenuItem aboutItem = new JMenuItem("About");
        aboutItem.addActionListener(this);
        fileMenu.add(aboutItem);
        JMenuItem exitItem = new JMenuItem("Exit");
        exitItem.addActionListener(this);
        fileMenu.add(exitItem);
    }

    public void actionPerformed(ActionEvent evt)
    {
        Object source = evt.getSource();
        if(source == aboutItem)
        {
            if (dialog == null) // first time
                dialog = new AboutDialog(this);
            dialog.show();
        }
        else if(source == exitItem)
        {
            System.exit(0);
        }
    }
}
```

```

    private AboutDialog dialog;
    private JMenuItem aboutItem;
    private JMenuItem exitItem;
}

class AboutDialog extends JDialog
{
    public AboutDialog(JFrame parent)
    {
        super(parent, "About DialogTest", true);

        Box b = Box.createVerticalBox();
        b.add(Box.createGlue());
        b.add(new JLabel("Core Java"));
        b.add(new JLabel("By Cay Horstmann and Gary Cornell"));
        b.add(Box.createGlue());
        getContentPane().add(b, "Center");

        JPanel p2 = new JPanel();
        JButton ok = new JButton("Ok");
        p2.add(ok);
        getContentPane().add(p2, "South");

        ok.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent evt)
            {
                setVisible(false);
            }
        });

        setSize(250, 150);
    }
}

public class DialogTest {
    public static void main(String[] args)
    {
        JFrame f = new DialogFrame();
        f.show();
    }
}

javax.swing.JDialog
● public JDialog(JFrame parent, String title, boolean modal)

        :      parent      가
          title
          modal      true

          가
        가      가      .

show()      가      .

```

9-45:

```

        ConnectInfo
        가
        가
        , transfer
        가
        ,
        ConnectDialog        showDialog
        .

        public boolean showDialog(ConnectInfo transfer)
        {
            username.setText(trnasfer.username);
            password.setText(transfer.password);
            ok = false; // set to true by Ok button action
            show();
            if (ok)
            {
                transfer.username = username.getText();
                transfer.password = password.getText();
            }
            return ok;
        }

        show()
        가 setVisible(false) dispose
        .( AWT .)

        가
        , windowClosing
        가
        transfer

        3 가
        . Ok
        , Cancel
        가 Ok
        ok
        true

        showDialog
        transfer . 가 true , Ok .

        ConnectInfo transfer = new ConnectInfo("yourname", "");
        if (dialog == null) dialog = new ConnectDialog(this);
        if (dialog.showDialog(transfer))
        {
            String uname = trnasfer.username
            String pwd = transfer.password;
        }

        :
        , show
        가
        .
        가
        Ok
        ,
        가
        가
        .
        (8 )
        .(2 javaBeans )
    
```

9-22

9-22: DataExchangeTest.java

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class DataExchangeTest extends JFrame
    implements ActionListener
{
    public DataExchangeTest()
    {
        setTitle("DataExchangeTest");
        setSize(300, 300);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        JMenuBar mbar = new JMenuBar();
        setJMenuBar(mbar);
        JMenu fileMenu = new JMenu("File");
        mbar.add(fileMenu);
        JMenuItem connectItem = new JMenuItem("Connect");
        connectItem.addActionListener(this);
        fileMenu.add(connectItem);
        JMenuItem exitItem = new JMenuItem("Exit");
        exitItem.addActionListener(this);
        fileMenu.add(exitItem);
    }

    public void actionPerformed(ActionEvent evt)
    {
        Object source = evt.getSource();
        if (source == connectItem)
        {
            ConnectInfo transfer
                = new ConnectInfo("yourname", "pw");
            if (dialog == null)
                dialog = new ConnectDialog(this);
            if (dialog.showDialog(transfer))
            {
                String uname = transfer.username;
                String pwd = transfer.password;
                Container contentPane = getContentPane();
                contentPane.add(new JLabel("username=" +
                    uname + ", password=" + pwd),
                    "South");
                validate();
            }
        }
        else if (source == exitItem)
            System.exit(0);
    }

    public static void main(String[] args)
    {
        JFrame f = new DataExchangeTest();
        f.show();
    }
}

```

```

    private ConnectDialog dialog = null;
    private JMenuItem connectItem;
    private JMenuItem exitItem;
}

class ConnectInfo
{
    public String username;
    public String password;
    public ConnectInfo(String u, String p)
    {
        username = u; password = p;
    }
}

class ConnectDialog extends JDialog
    implements ActionListener
{
    public ConnectDialog(JFrame parent)
    {
        super(parent, "Connect", true);
        Container contentPane = getContentPane();
        JPanel p1 = new JPanel();
        p1.setLayout(new GridLayout(2, 2));
        p1.add(new JLabel("User name:"));
        p1.add(username = new JTextField(""));
        p1.add(new JLabel("Password:"));
        p1.add(password = new JPasswordField(""));
        contentPane.add("Center", p1);

        Panel p2 = new Panel();
        okButton = addButton(p2, "Ok");
        cancelButton = addButton(p2, "Cancel");
        contentPane.add("South", p2);
        setSize(240, 120);
    }

    JButton addButton(Container c, String name)
    {
        JButton button = new JButton(name);
        button.addActionListener(this);
        c.add(button);
        return button;
    }

    public void actionPerformed(ActionEvent evt)
    {
        Object source = evt.getSource();
        if(source == okButton)
        {
            ok = true;
            setVisible(false);
        }
        else if (source == cancelButton)
            setVisible(false);
    }

    public boolean showDialog(ConnectInfo transfer)
    {
        username.setText(transfer.username);
        password.setText(transfer.password);
        ok = false;
        show();
    }
}

```

```

        if (ok)
        {
            transfer.username = username.getText();
            transfer.password = password.getText();
        }
        return ok;
    }

    private JTextField username;
    private JTextField password;
    private boolean ok;
    private JButton okButton;
    private JButton cancelButton;
}

```

```

        ,
        ,
        .
        .
        Swing 가
        JFileChooser
        JFileChooser
        JDialog
        JFileChooser
        가
        show
        showSaveDialog
        showSaveDialog
        Open
        save 가
        showDialog
        9-46

```

9-46:

가

1. JFileChooser . JDialog

```

JFileChooser d = new JFileChooser();

```

2. setCurrentDirectory

```

d.setCurnnetDictionary(new File("."));

```

```

. File 12 File 가
File File(String ) 가

```

```

3.      가
   setSelectedFile
      .

   d.setSelectedFile(new File(filename));

4.      가
   setMultiSelectionEnabled
      .

   d.setMultiSelectedEnabled(true);

5.      , .gif      가      )      (file filter)      (
      .

6. showOpenDialog      showSaveDialog      가
   .

   int result = d.showOpenDialog(parent);

   int result = d.showSaveDialog(parent);

      가
      .      JFileChooser.APPROVE_OPTION
JFileChooser.CANCEL_OPTION      .
7.      setSelectedFile()      getSelectedFiles()
      File      File
      , getName      .

   String filename = d.getSelectedFile().getName();

      .      가
      .      .gif      ,      가 GIF
      .      "GIF      "      가
      .      가 JPEG      .jpg      .jpeg      가

      가
      javax.swing.filechooser.FileFilter      .

      .      가

      .      ,      FileFilter

      가      .      ,

      가      .      ExtensionFileFilter

   ExtensionFileFilter filter = new ExtensionFileFilter();
   filter.addExtension("jpg");
   filter.addExtension("gif");
   filter.setDescription("JPG & GIF Image");

```

JDK 가
 FileFilter 2

```
public boolean accept(File f);
public String getDescription();
```

가
 GIF

```
public class GifFilter implements FileFilter
{
    public boolean accept(File f)
    {
        return f.getName().toLowerCase().endsWith(".gif") ||
            f.isDirectory();
    }
    public String getDescription()
    {
        return "GIF Image";
    }
}
```

가
 JFileChooser setFileFilter

```
d.setFileFilter(new GifFilter());
```

```
, ,
d.setFileFilter(new FileFilter
{
    public boolean accept(File f)
    {
        return f.getName().toLowerCase().endsWith(".gif") ||
            f.isDirectory();
    }
    public String getDescription()
    {
        return "GIF Image";
    }
} );
```

, 가

. 가 가 addChoosableFileFilter
 . , getAcceptAllFileFilter

가 가

```
d.addChoosableFileFilter(d.getAcceptableAllFileFilter());
```

가

: java.io FileFilter
 boolean accept(File f)가 File listFile ,

```

        Swing
        가 Sun

java.io javax.swing.filechooser
        가
javax.swing, filechooser.FileFilter javax.swing, filechooser.*
(import)

```

```

        가
        javax.swing.filechooser
FileView 가
        , look feel
        가 ,
        FileView 5

Icon getIcon(File f);
String getName(File f);
String getDescription(File f);
String getTypeDescription(File f);
Boolean isTraversable(File f);

        setFileView

가 , , null look feel
        가
        가
        isTraversable Boolean
Boolean !
        null
        3가
        Boolean true(Boolean.ture) false
(Boolean.false) (null) 가
        가 .java 가
        ,

class CoffeeIconFileView extends javax.swing.fiilechooser.FileView
{
    public Icon getIcon(File f)
    {
        if (f.getName().toLowerCase().endsWith(".java"))
            return new ImageIcon("coffee.gif");
        else
            return null;
    }
    public String getDescription(File f) { return null; }
    public String getName(File f) { return null; }
    public String getTypeDescription(File f) { return null; }
    public Boolean isTraversable (File f) { return null; }
}

```

```
}
```

```
setFileView
```

```
chooser.setFileView(new CoffeeIconFileView());
```

```
JDK demo/jfc/FileChooserDemo
```

```
ExampleFiileView
```

```
javax.swing.JFileChooser
```

- JFileChooser()

- void setCurrentDirectory(File dir)

- setSelectedFile(File file)

- setSelectedFile(File[] file)

- void setMultiSelectionEnabled(boolean b)

- int showOpenDialog(Component parent)

- int showSaveDialog(Component parent)

```
"File open" "File save"
```

```
. APPROVE_OPTION
```

```
CANCEL_OPTION
```

- File getFile()

- File[] getFiles()

```
가
```

```
.(
```

```
가
```

```
null
```

```
.)
```

- void setFileFilter(javax.swing.filechooser.FileFilter filter)

```
. filter.accept가 true
```

- void setFileView(FileView view)

```
가
```

```
javax.swing.filechooser.FileFilter
```

- boolean accept(File f)

```
가
```

```
true
```

- String getDescription()

```
"Image files (*.gif, *.jpeg)"
```

javax.swing.filechooser.FileView

- String getName(File f)
f null . f.getName()
.
- String getDescription(File f)
f null . , f가 HTML
, "Hypertext document"
.
- String getTypeDescription(File f)
f null . , f가 HTML
, "Hypertext document" .
- Icon getIcon(File f)
f null . , f가 JPEG ,
가 .
- Boolean isTraversable(File f)
가 Boolean.TRUE . 가
false . FileView ,
가 null ,
.