第11章

Swing基本组件

本章相关词汇

单　　词	说　　明
dialog	会话，对话框
scroll	卷轴，滚动
password	密码
area	区域，面积
checkBox	复选框
radio	单选按钮
comboBox	组合框
group	团体，组
font	字体
wrap	包装，缠绕

本章目标

- 了解AWT以及java.awt包
- 了解Swing组件和javax.swing包
- javax.swing包中的常用组件：
  1. 容器组件
     ① JFrame
     ② JPanel
  2. 文本组件
     ① JButton、JLabel
  3. 表单组件
     ① JTextField、

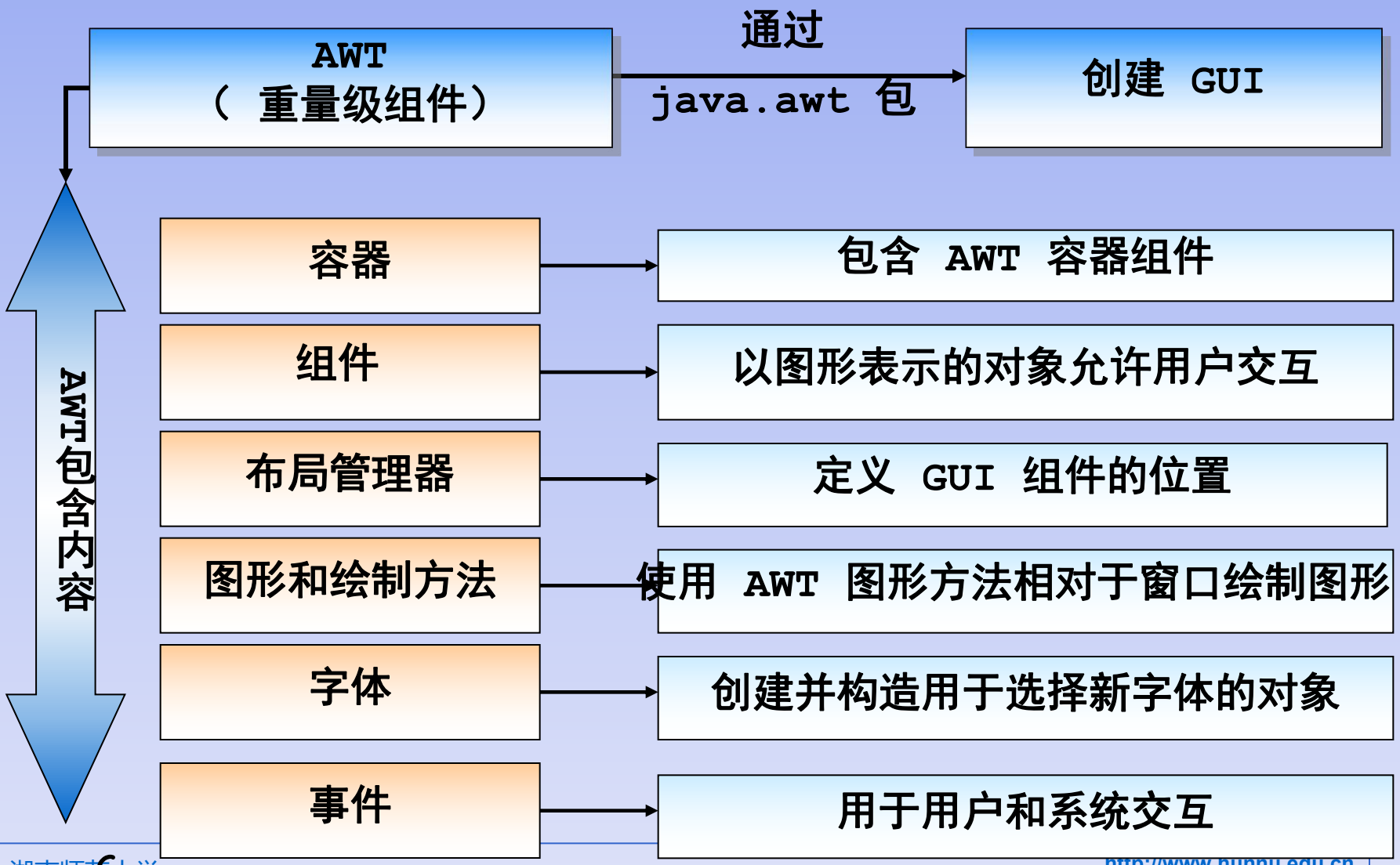
GUI的概念

- 到目前为止，我们在C和Java中编写的都是基于控制台的程序；
- GUI（Graphical User Interface）即图形用户界面，它能够使应用程序看上去更加友好；
- Java语言之所以如此流行的一个主要原因，就是因为它支持GUI；

AWT简介

- 实现GUI编程是由一系列图形化组件来完成的（即一系列定义好的类），这些组件也被称为控件；
- 在Java的早期版本中，GUI组件由名为AWT（Abstract Window Toolkit，抽象窗口工具包）的标准库来提供；
- 除了GUI组件外，AWT还包括其它功能来支持图像绘画、处理剪切/复制类型的数据传送，以及其它相关操作。

AWT 概述

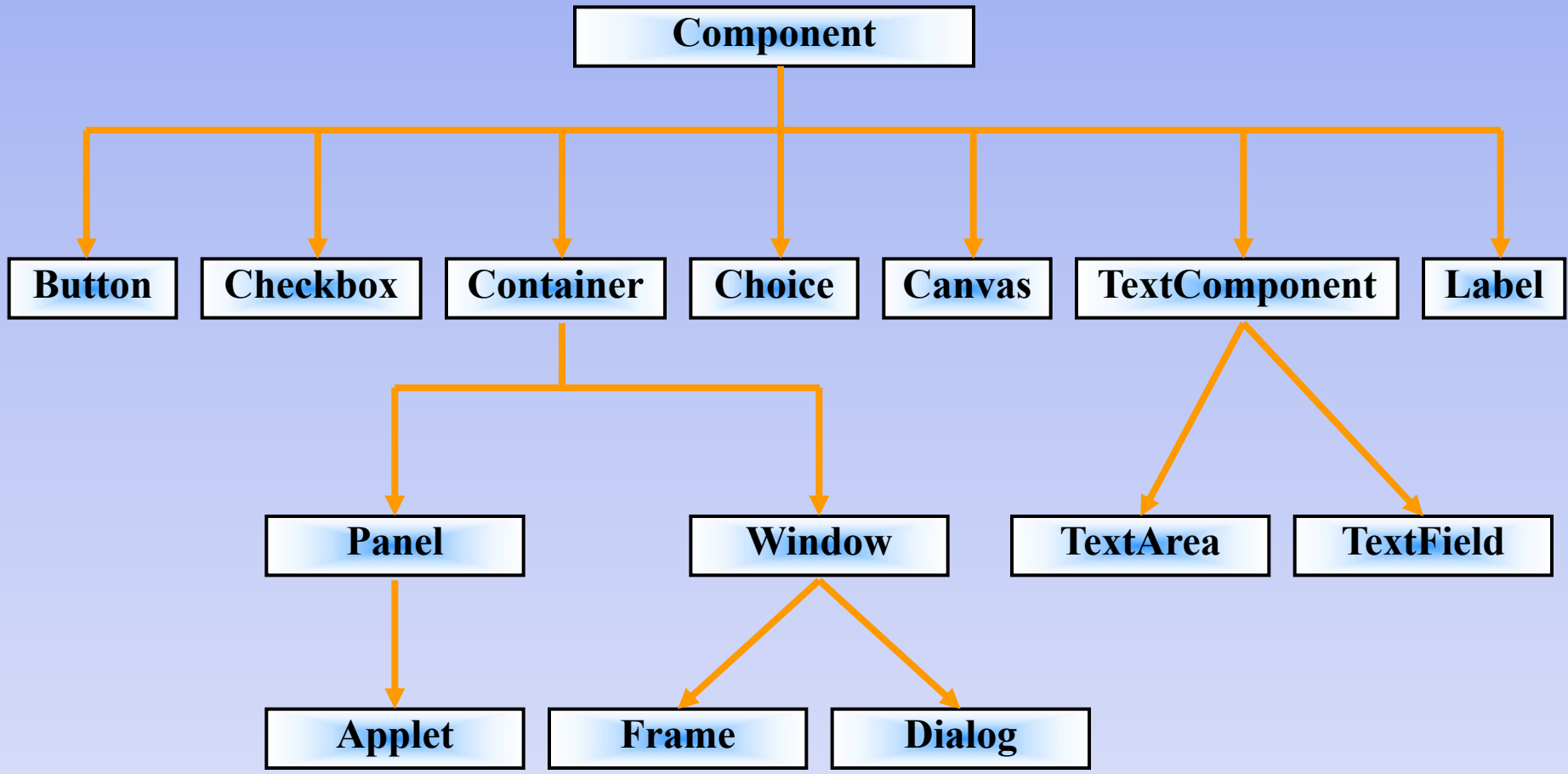

AWT
（ 重量级组件）
通过
java.awt 包
创建 GUI
AWT包含内容
容器
包含 AWT 容器组件
组件
以图形表示的对象允许用户交互
布局管理器
定义 GUI 组件的位置
图形和绘制方法
使用 AWT 图形方法相对于窗口绘制图形
字体
创建并构造用于选择新字体的对象
事件
用于用户和系统交互

java.awt包

- java.awt包是Java内置的包，属于Java基础类库（JFC）的一部分，其中包括以下内容：
  1. 便于用户输入的一组丰富的界面组件；
  2. 将组件放置在适当位置的几种布局管理器；
  3. 事件处理模型；
  4. 图形和图像工具等等。
- 要使用到该包中的类，则必须显式地声明如下语句：

```
import java.awt.*;
```

AWT组件的类体系结构



AWT编程示例

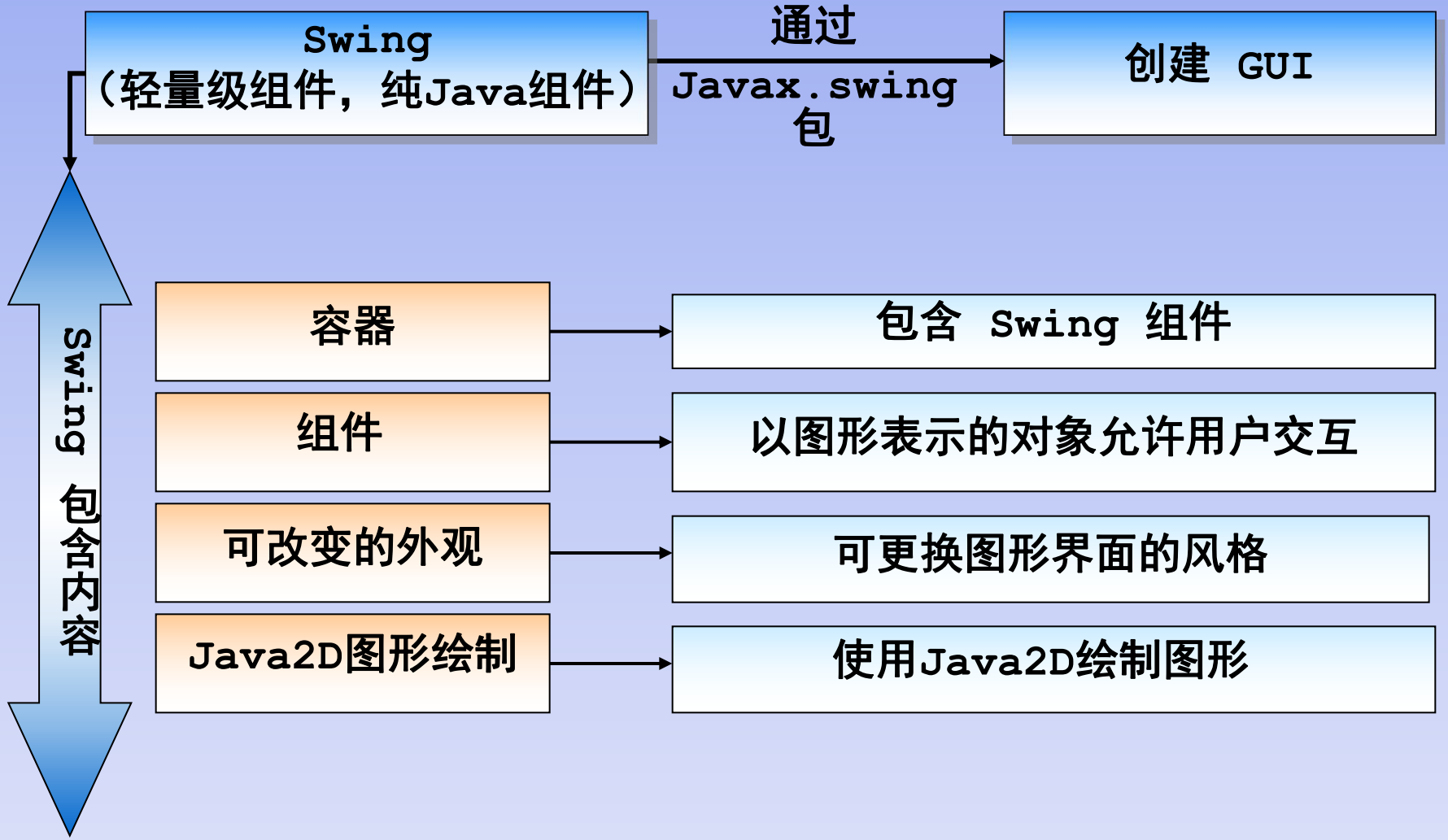
- AWT组件最大的缺陷是它依赖于操作系统，也就是说，AWT程序运行在不同的操作上会有不同的外观和行为，这一点对于Java的平台无关性来讲，是无法容忍的。

Swing简介以及javax.swing包

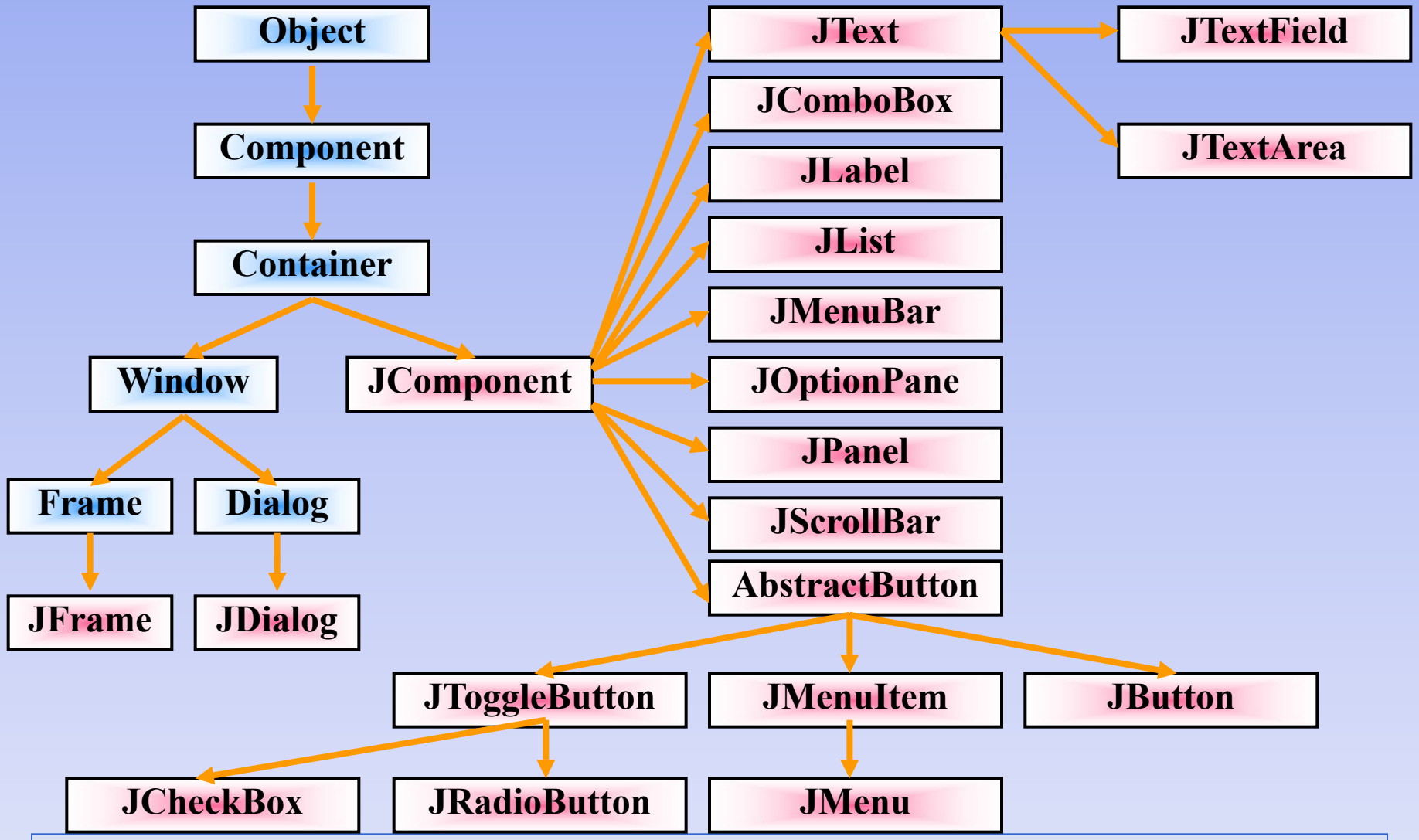
- Swing组件是在AWT组件基础上发展而来的轻量级组件，与AWT相比不但改进了用户界面，而且所需的系统资源更少；
- Swing是纯Java组件，使得应用程序在不同的平台上运行时具有相同外观和相同的行为。
- javax.swing包中包含了一系列Swing组件，如果要使用该包中的类，则必须显式地声明如下语句：

```
import javax.swing.*;
```

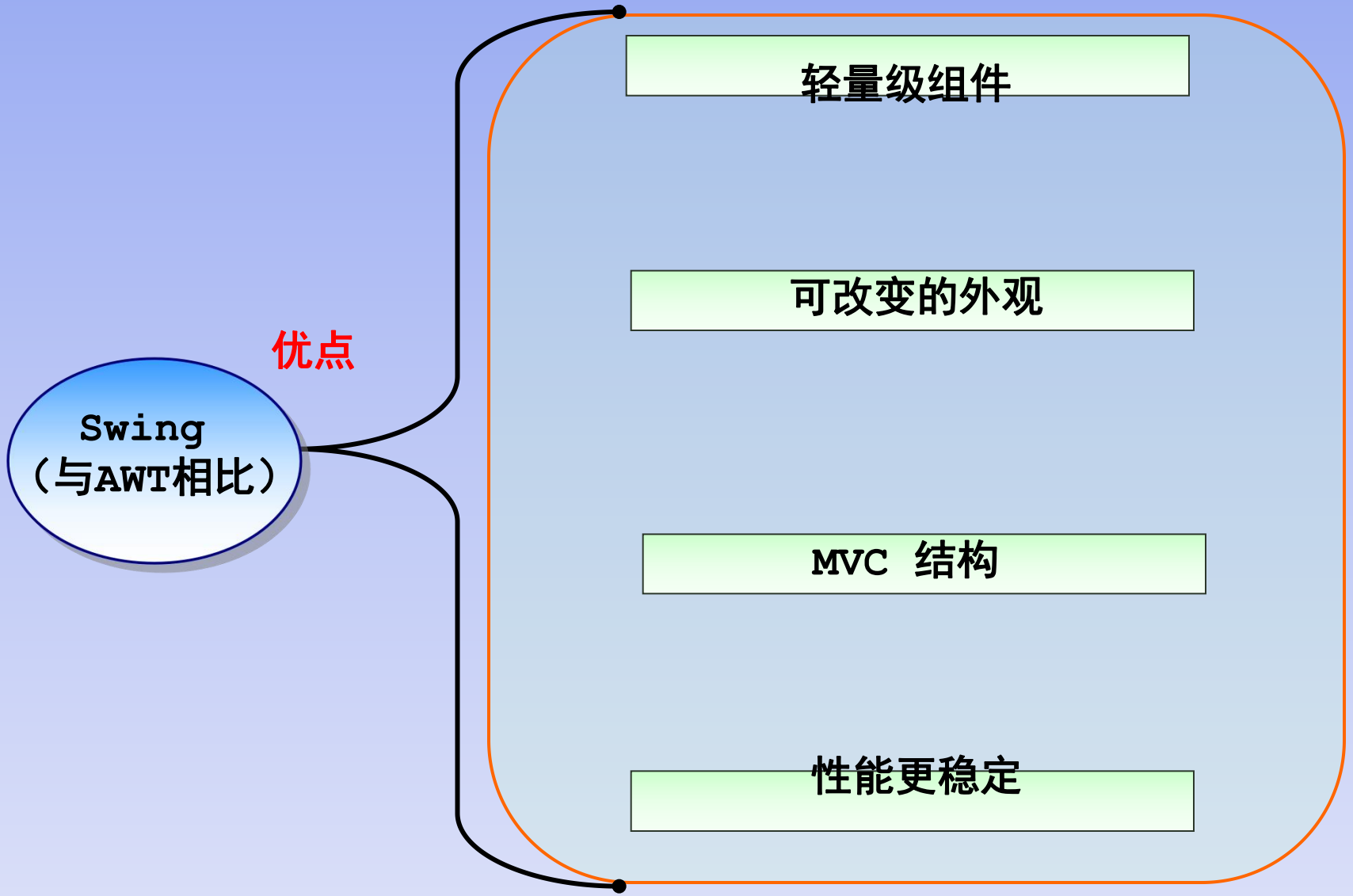
Swing 2-1


Swing
（轻量级组件，纯Java组件）
通过
Javax.swing
包
创建 GUI
Swing 包含内容
容器
包含 Swing 组件
组件
以图形表示的对象允许用户交互
可改变的外观
可更换图形界面的风格
Java2D图形绘制
使用Java2D绘制图形

Swing组件的类体系结构



Swing 2-2



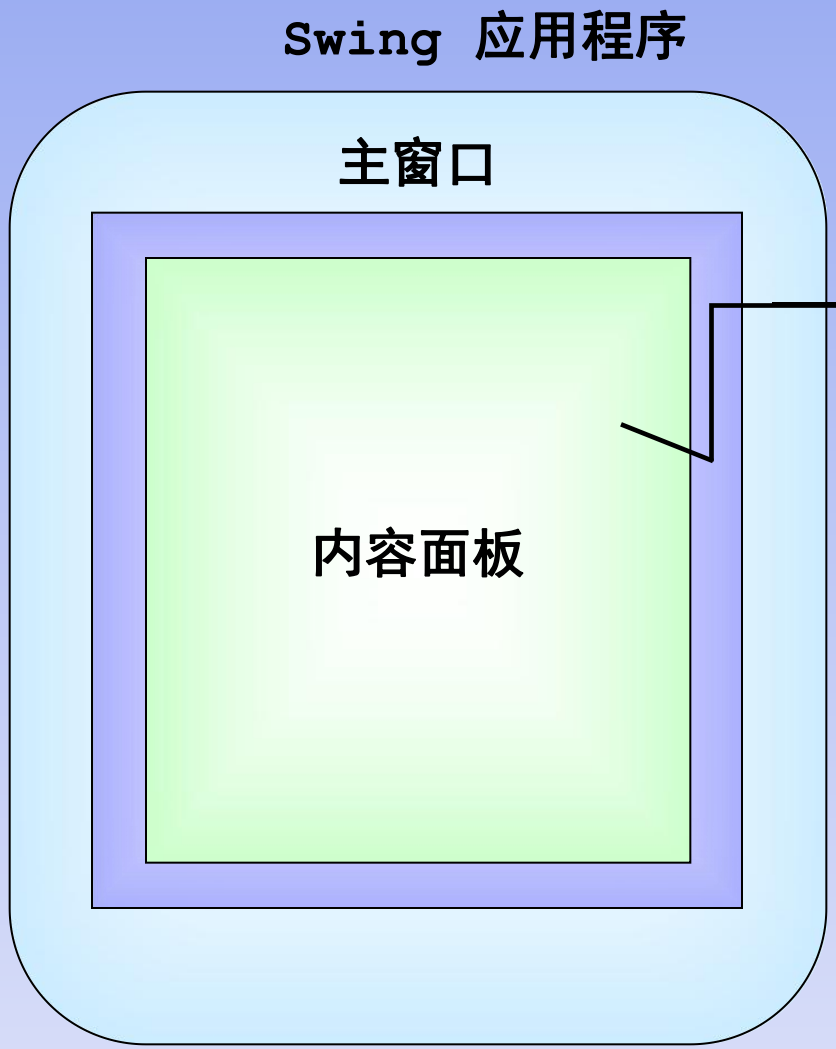
常用Swing组件

- 我们将常用的Swing组件根据其性质不同，分类进行介绍，其中包括：
  1. 容器组件
  2. 文本组件
  3. 表单组件

Swing中常用的容器组件

- 容器组件是指可以容纳其它组件的组件，常用的Swing容器包括：
  1. JFrame（框架）
  2. Container（容器）
  3. JDialog（对话框）
  4. JPanel（面板）
  5. JScrollPane（滚动面板）

Swing 容器组件 3-1



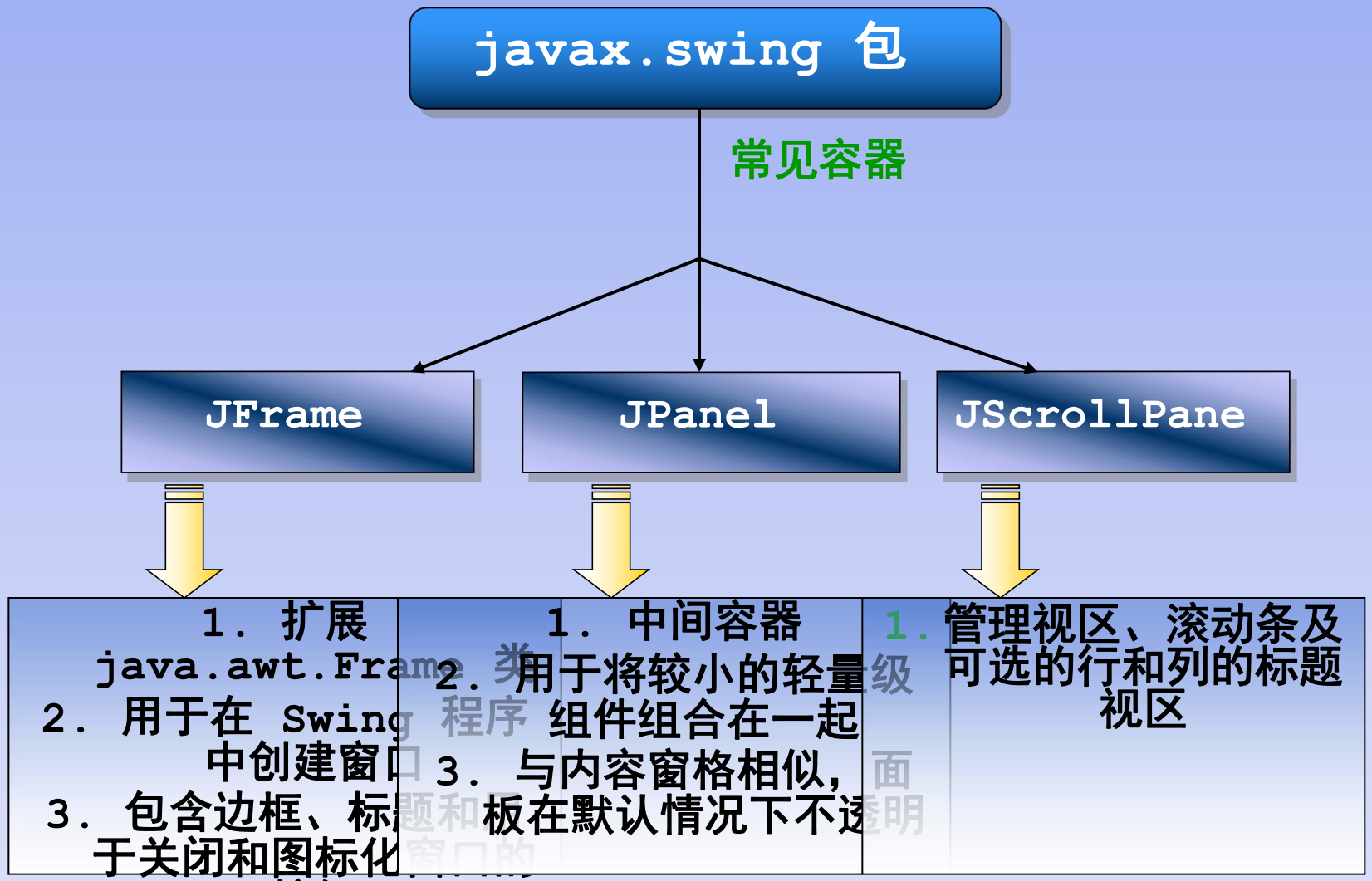
顶层容器（每个应用程序中至少有一个）

主要显示区域包含 GUI 中的可视组件

创建顶层容器

- 创建容器
- 设置容器大小（对框架及面板而言）
- 设置容器可见度（对框架及面板而言）

Swing 容器组件 3-2



javax.swing.JFrame

- JFrame组件用于在Swing程序中创建窗体；
- JFrame类的构造方法有4种重载方式，以下是常用的几种：

构 造 方 法	说　　明
JFrame()	创建新窗体，该窗体初始为不可见
JFrame(String title)	创建新窗体，使用参数title指定标题，该窗体初始为不可见

JFrame的常用方法

方 法 原 型	说　明
void setTitle(String title)	设置窗体的标题，标题内容由参数title指定
void setSize(int width, int height)	设置窗体的大小，参数width指定宽度，参数height指定高度，单位是像素
void setResizable(boolean resizable)	设置窗体能否调整大小，由参数resizable决定
void setVisible(boolean b)	设置窗体是否为可见，由参数b决定，true为可见，false为不可见
Container getContentPane()	获得当前窗体的内容面板
void setDefaultCloseOperation(int operation)	设置窗体在关闭时默认执行的操作
void dispose()	释放当前窗体及其所有子组件所占用的资源，即卸载窗体
void repaint()	重新绘制当前窗体

创建窗体

- 对于类似于窗体这样的容器组件，我们一般自定义一个类，继承于JFrame类，然后将窗体中的子组件作为类中成员进行声明，以方便操作，如：

```
public class MyFrame extends JFrame
{
    ……
}
```

- 容器组件是指可以容纳其它组件的组件。

创建窗体示例


```
import javax.swing.*;  //导入必要的包

/**自定义窗体类，继承于JFrame类*/
public class MyFrame extends JFrame
{
    /**构造方法*/
    public MyFrame()
    {
        //super("这是我的第一个窗体");          //利用父类的构造方法设置标题
        this.setTitle("这是我的第一个窗体"); //设置窗体的标题
        this.setSize(300, 200);              //设置窗体的大小
        this.setVisible(true);               //设置窗体为可见，即显示窗体
    }

    /**main方法，程序入口*/
    public static void main(String[] args)
    {
        MyFrame mf = new MyFrame();  //创建窗体实例
    }
}
```

窗体的内容面板

- 事实上，一个完整的窗体是由外部框架和内容面板两部分组成的；
- 外部框架是指由标题栏和四边所组成空心边框，它主要用来控制窗体的大小和外观；
- 我们实际操作的是内容面板，如设置窗体的背景色，设置窗体的布局，往窗体中添加其它组件等等；
- 使用getContentPane方法获得当前窗体的内容面板，该方法的返回值是Container（容器）类对象，如：

```
Container contentPane = getContentPane();
```

- Container类在java.awt包中。

java.awt.Container

- Container类通常用于操作JFrame的内容面板，其常用的方法有：

方 法 原 型	说　明
void setBackground(Color bg)	设置容器的背景色，由参数bg指定颜色
void setLayout(LayoutManager mgr)	设置容器的布局，参数是布局管理器
Component add(Component comp)	往容器中添加一个组件
Component add(Component comp, int index)	将指定组件添加到容器中的指定位置上
void remove(Component comp)	从容器中移除指定的组件
void removeAll()	从容器中移除所有组件
void repaint()	重新绘制当前容器

内容面板示例

```
import java.awt.*;  //Container类和Color类在此包中
import javax.swing.*;

public class ContentPaneDemo extends JFrame {
    public ContentPaneDemo() {
        super("内容面板示例");
        //获得当前窗体的内容面板
        Container contentPane = this.getContentPane();
        //设置内容面板的背景色为红色
        contentPane.setBackground(Color.RED);
        //设置窗体关闭时即退出程序
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 200);
        setResizable(false);  //设置窗体不可调整大小
        setVisible(true);
    }

    public static void main(String[] args) {
        ContentPaneDemo cpd = new ContentPaneDemo();
    }
}
```

java.awt.Color

- Color类用于创建颜色对象，其构造方法共有7种重载方式，以下是常用的几种：

构 造 方 法	说　　明
Color(int r, int b, int g)	使用指定的红、蓝、绿的色值创建sRGB颜色对象，这些值都在0~255的范围之内

常　量	颜　色	常　量	颜　色
Color.BLACK	黑色	Color.BLUE	蓝色
Color.CYAN	青色	Color.DARK_GRAY	深灰色
Color.GRAY	灰色	Color.GREEN	绿色
Color.LIGHT_GRAY	浅灰色	Color.MAGENTA	洋红色
Color.ORANGE	桔黄色	Color.PINK	粉红色
Color.RED	红色	Color.WHITE	白色
Color.YELLOW	黄色		

javax.swing.JPanel

- JPanel提供面板组件，它是轻量级的容器组件；
- 面板中可以添加其它组件，也可以设置布局，我们一般使用面板来实现布局嵌套；
- JPanel类的构造方法有4种重载方式，以下是常用的几种：

构 造 方 法	说　　明
JPanel()	创建一个空面板
JPanel(LayoutManaer layout)	创建带有指定布局的面板

JPanel的常用方法

- JPanel（面板）的操作方式与Container（内容面板）很相似，以下是一些常用方法：

方 法 原 型	说　明
void setBackground(Color bg)	设置面板的背景色，由参数bg指定颜色
void setLayout(LayoutManager mgr)	设置面板的布局，参数是布局管理器
Component add(Component comp)	往面板中添加一个组件
Component add(Component comp, int index)	将指定组件添加到面板中的指定位置上
void remove(Component comp)	从面板中移除指定的组件
void removeAll()	从面板中移除所有组件
void repaint()	重新绘制当前面板

javax.swing.JScrollPane

- JScrollPane是滚动面板组件，当某些组件的可视区域不足以显示其全部内容时，可以将该组件添加到滚动面板中，为其增加滚动条。

JScrollPane的构造方法

- JScrollPane的构造方法共有4种重载：

构 造 方 法	说　　明
JScrollPane()	创建一个空的JScrollPane，需要时水平和垂直滚动条都可显示
JScrollPane(Component view)	创建一个显示指定组件内容的 JScrollPane，只要组件的内容超过视图大小就会显示水平和垂直滚动条
JScrollPane(Component view, int vsbPolicy, int hsbPolicy)	创建一个显示指定组件内容的 JScrollPane ，并指定滚动条的显示策略
JScrollPane(int vsbPolicy, int hsbPolicy)	创建一个空的JScrollPane，并指定滚动条的显示策略

JScrollPane的常用方法

方 法 原 型	说　　明
JScrollBar getHorizontalScrollBar()	返回当前滚动面板的水平滚动条
JScrollBar getVerticalScrollBar()	返回当前滚动面板的垂直滚动条
JViewport getViewport()	返回当前滚动面板的 JViewport

布局管理器

- 在MyEclipse中，组件的定位是靠坐标来完成的，而在手工编码中，坐标定位将是非常麻烦的工作；
- 用户界面上的组件可以按照不同的方式进行排列，例如：可以依序水平排列，或者按网格方式进行排列；
- 每种方案都是指组件的一种布局，要管理这些布局，就需要使用布局管理器；
- 布局管理器是一组实现了java.awt.LayoutManager接口的类，由这些类自动定位组件；
- 布局管理器类在java.awt包中。

布局管理器

- 布局管理器用来：
  1. 决定组件在容器上如何摆放；
  2. 决定组件的大小；
- Frame的默认布局管理器是FlowLayout；
- JAVA中常用的布局：
  1. FlowLayout（流式布局）
  2. BorderLayout（边框布局）
  3. GridLayout（网格布局）
  4. CardLayout （卡片布局）

几种常用布局

流式布局

java.awt.FlowLayout

边界布局

java.awt.BorderLayout

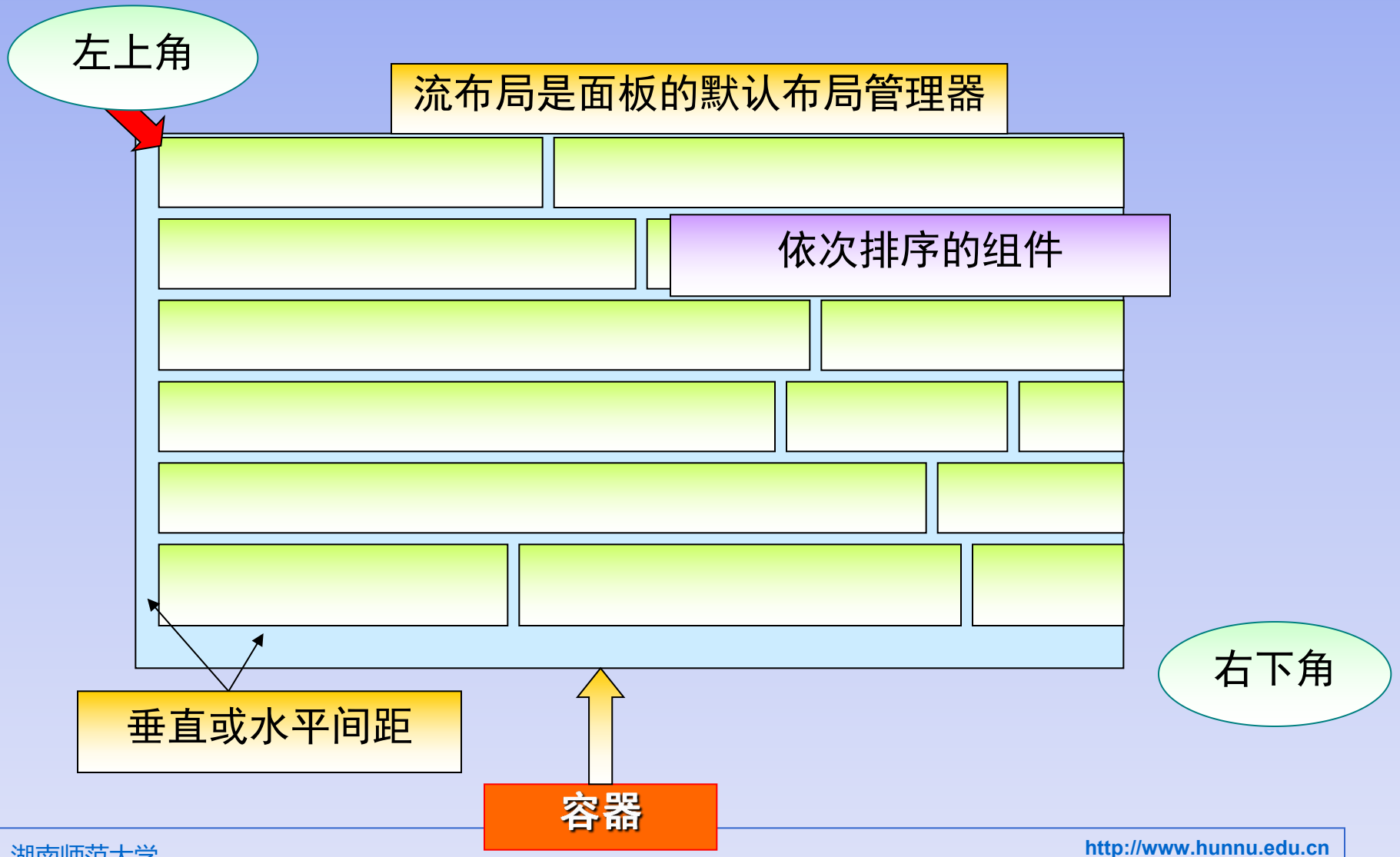
网格布局

java.awt.GridLayout

为容器设置布局

- 布局管理器(LayoutManager)指的是FlowLayout,BorderLayout等类的对象；
- 调用容器对象的setLayout(LayoutManager lm)方法，即可为容器设置不同的布局；
- 利用多种复杂布局的组合，总可以达到你想要的效果

FlowLayout

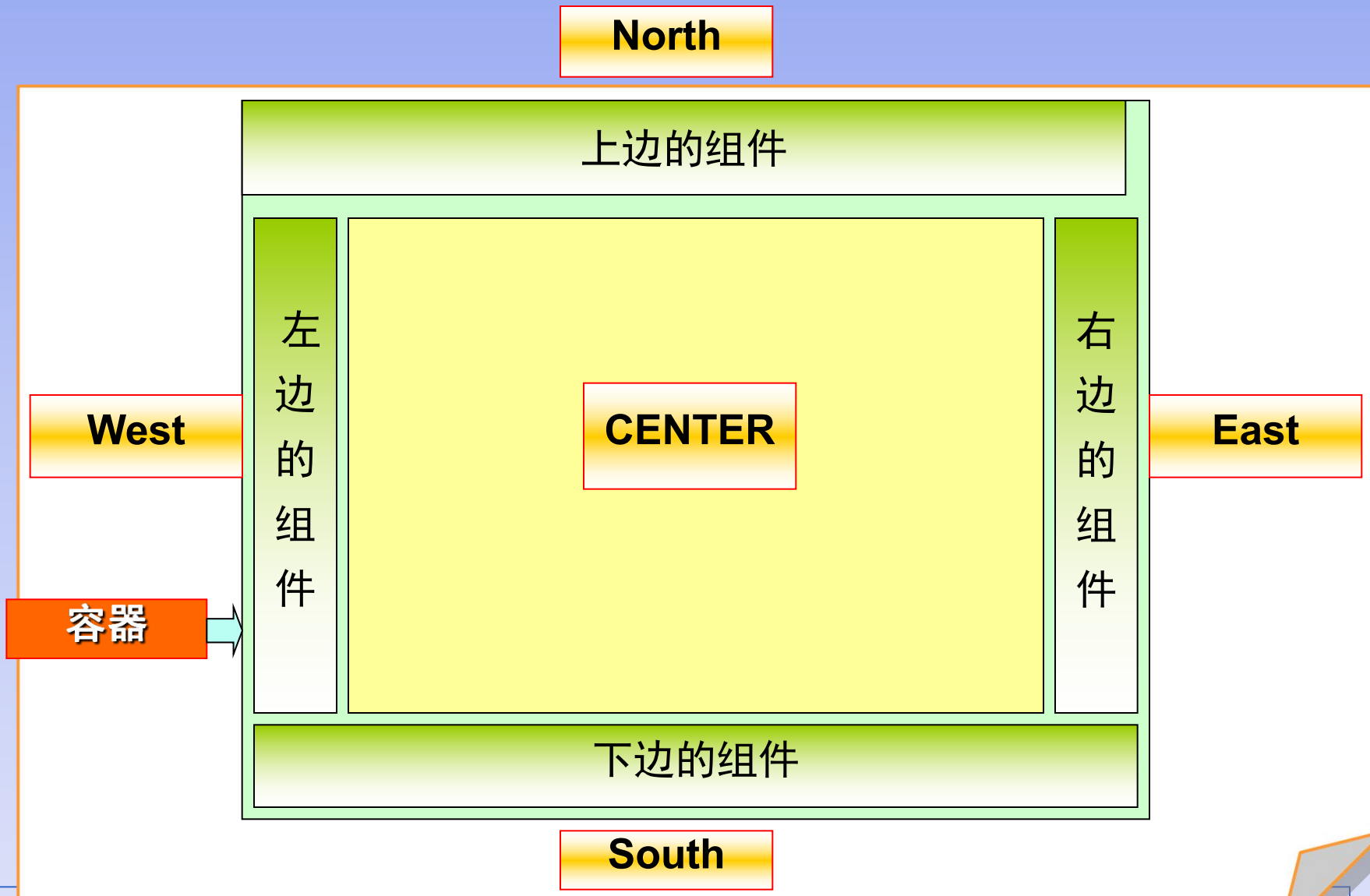


流式布局示例

```
import java.awt.*;      //布局管理器在此包中
import javax.swing.*;

public class FlowLayoutDemo extends JFrame {
    private JTextField txt1, txt2, txt3, txt4;  //声明4个文本框
    public FlowLayoutDemo() {
        super("流式布局示例"); //使用父类的构造方法设置窗体标题
        txt1 = new JTextField("文本框1"); //分别实例化4个文本框
        txt2 = new JTextField("文本框2");
        txt3 = new JTextField("文本框3");
        txt4 = new JTextField("文本框4");
        Container me = getContentPane(); //获取当前窗体的内容面板
        me.setLayout(new FlowLayout());  //设置内容面板的布局为流式布局
        me.add(txt1);     //分别将4个文本框添加到内容面板中
        me.add(txt2);
        me.add(txt3);
        me.add(txt4);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 200);
        setVisible(true);
    }
    public static void main(String[] args) {
        new FlowLayoutDemo();  //实例化窗体，匿名对象
    }
}
```

BorderLayout



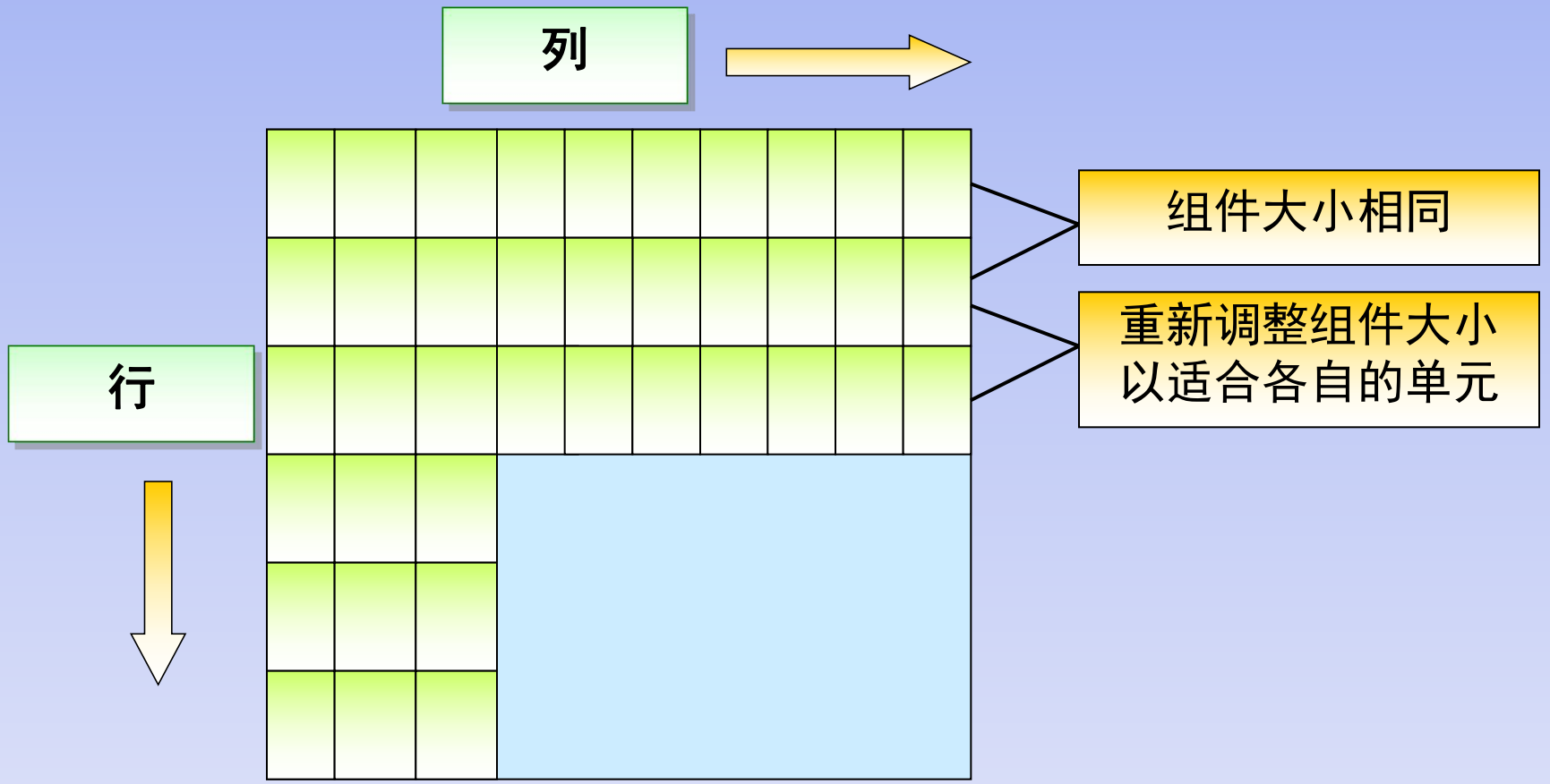
边界布局示例

```
public class BorderLayoutDemo extends JFrame {
    private JButton btn1, btn2, btn3, btn4, btn5;  //声明5个按钮
    public BorderLayoutDemo() {
        btn1 = new JButton("北边按钮"); //分别实例化5个按钮
        btn2 = new JButton("南边按钮");
        btn3 = new JButton("西边按钮");
        btn4 = new JButton("东边按钮");
        btn5 = new JButton("中间按钮");
        Container me = getContentPane();
        me.setLayout(new BorderLayout());   //设置内容面板的布局为边界布局
        me.add(btn1, BorderLayout.NORTH);   //分别将按钮添加到内容面板的各个方位
        me.add(btn2, BorderLayout.SOUTH);
        me.add(btn3, BorderLayout.WEST);
        me.add(btn4, BorderLayout.EAST);
        me.add(btn5, BorderLayout.CENTER);
        setTitle("边界布局示例");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 200);
        setVisible(true);
    }

    public static void main(String[] args) {
        new BorderLayoutDemo();
    }
}
```

GridLayout

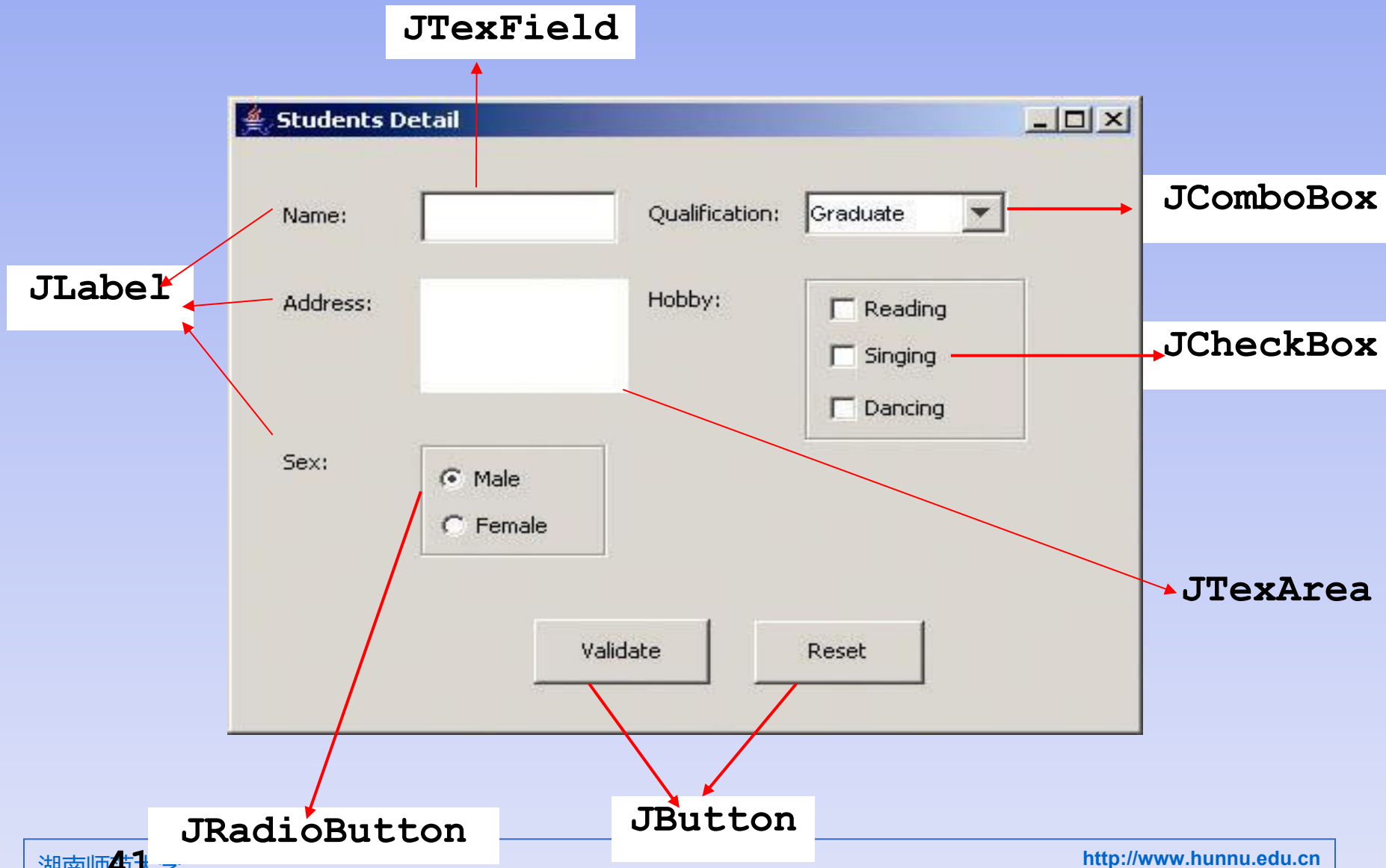
指定网格中的行数和列数，创建网格布局



CardLayout

- 可存储几个不同的布局。
- 每个布局就像是一个卡片组中的一张卡片。
- 在一个给定的时间总会有一张卡片在顶层。
- 卡片通常为一个 Panel 对象。
- 每当需要许多面板切换，而每个面板需要显示为不同布局时，可以使用卡片布局

Swing GUI 组件


JTexField
Students Detail
Name:
Qualification:
Graduate
JComboBox
JLabel
Address:
Hobby:
Reading
Singing
Dancing
JCheckBox
Sex:
Male
Female
JTexArea
Validate
Reset
JRadioButton
JButton

Swing中常用的文本组件

- 所谓文本组件是指专门用来存放文字的组件，包括：
  1. Jbutton（按钮）
  2. JLabel（标签）

javax.swing.JButton

- 在Swing程序中，按钮可能是使用量最大的组件之一，JButton则是用来创建按钮的；
- JButton类的构造方法有5种重载方式，以下是常用的几种：

构 造 方 法	说　　明
JButton()	创建一个空按钮
JButton(String text)	创建一个带文本的按钮
JButton(Icon icon)	创建一个带图标的按钮
JButton(String text, Icon icon)	创建一个带文本和图标的按钮

JButton的常用方法

方 法 原 型	说　明
void setText(String text)	设置按钮上的文本
String getText()	获得按钮上的文本
void setBackground(Color bg)	设置按钮的背景色
Color getBackground()	获得按钮的背景色
void setEnabled(boolean b)	设置启用（或禁用）按钮，由参数b决定
void setVisible(boolean b)	设置按钮是否为可见，由参数b决定
void setToolTipText(String text)	设置按钮的悬停提示信息
void setMnemonic(int mnemonic)	设置按钮的快捷键

javax.swing.JLabel

- JLabel是最简单的Swing组件之一，用于在窗体上显示标签， JLabel既可以显示文本，也可以显示图像；
- JLabel类的构造方法有6种重载方式：

构 造 方 法	说　　明
JLabel()	创建一个空的标签
JLabel(String text)	创建一个带文本的标签
JLabel(String text, int ha)	创建一个带文本的标签，并指定其对齐方式，可以是JLabel.LEFT、JLabel.CENTER和JLabel.RIGHT
JLabel(Icon image)	创建一个带图像的标签
JLabel(Icon image, int ha)	创建一个带图像的标签，并指定其对齐方式
JLabel(String text, Icon image, int ha)	创建一个带文本和图像的标签，并指定其对齐方式

JLabel的常用方法

方 法 原 型	说　明
void setText(String text)	设置标签上的文本
String getText()	获得标签上的文本
void setIcon(Icon icon)	设置标签中的图像
Icon getIcon()	获得标签中的图像
void setHorizontalAlignment(int alignment)	设置标签中文本的对齐方式
void setVisible(boolean b)	设置标签是否为可见

Swing中常用的表单组件

- 这里套用了HTML中的“表单”一词，常用的表单组件包括：
  1. JTextField（文本框）
  2. JPasswordField（密码框）
  3. JTextArea（文本域）
  4. JCheckBox（复选框）
  5. JRadioButton（单选按钮）
  6. JComboBox（组合框，又名：下拉列表）

javax.swing.JTextField

- JTextField是文本框组件，主要用来接受用户的输入；
- JTextField类的构造方法有5种重载方式，以下是常用的几种：

构 造 方 法	说　　明
JTextField()	创建一个空的文本框
JTextField(String text)	创建一个带文本的文本框
JTextField(int columns)	创建一个指定列数的空文本框
JTextField(String text, int columns)	创建一个带文本，并指定列数的文本框

JTextField的常用方法

方 法 原 型	说　明
void setText(String text)	设置文本框中的文本
String getText()	获得文本框中的文本
void setHorizontalAlignment(int alignment)	设置文本框中文本的对齐方式，可以是JTextField.LEFT、JTextField.CENTER和JTextField.RIGHT
void setEditable(boolean b)	设置文本框是否可以编辑，由参数b决定
void setEnabled(boolean enabled)	设置启用（或禁用）文本框
void setVisible(boolean b)	设置文本框是否为可见

javax.swing.JPasswordField

- JPasswordField用来提供密码框组件，它的构造方法共有5种重载，以下是常用的几种：

构 造 方 法	说　　明
JPasswordField()	创建一个空的密码框
JPasswordField(String text)	用指定文本初始化密码框
JPasswordField(int columns)	创建一个指定列数的空密码框
JPasswordField(String text, int columns)	创建一个带文本，并指定列数的密码框

JPasswrodField的常用方法

方 法 原 型	说　　明
void setText(String text)	设置密码框中的文本
String getText()	获得密码框中的文本，出于安全考虑，此方法已过时，由getPasswrod方法替代
char[] getPassword()	获得密码框中的文本，只不过是以字符数组的方式返回
void setEchoChar(char c)	设置密码框的密文字符

javax.swing.JTextArea

- 当用户有大量文本需要输入的时候，就可以使用到文本域组件，JTextArea的构造方法共有6种重载，以下是常用的几种：

构 造 方 法	说　　明
JTextArea()	创建一个空的文本域
JTextArea(String text)	用指定文本初始化文本域
JTextArea(int rows, int columns)	创建一个指定行数和列数的空文本域
JTextArea(String text, int rows, int columns)	创建一个带文本，并指行数和列数的文本域

JTextArea的常用方法

方 法 原 型	说　　明
void setText(String text)	设置文本域中的文本
String getText()	获得文本域中的文本
void setFont(Font font)	设置文本域中文本的字体
void setLineWrap(boolean wrap)	设置文本域中文本的自动换行策略
void setTabSize(int size)	设置制表符 ‘\t’所占的字符宽度，默认为8个字符宽度

java.awt.Font

- Font类用来表示字体，常用的构造方法如下：

构 造 方 法	说　　明
Font(String name, int style, int size)	构造一个Font对象，参数name指定字体名称，style指定字体样式（可以是Font.BOLD、Font.ITALIC和Font.PLAIN），size指定字体的大小

用Font.BOLD + Font.ITALIC表示，Font.PLAIN表示普通样式；

- 任何包含有文字的组件，都可以使用setFont方法来设置字体。

javax.swing.JCheckBox

- JCheckBox用来提供复选框组件，一般用来提供多个选项，并且可选项不限定的情况下，可以使用到复选框；
- 其构造方法共有8种重载，常用的如下：

构 造 方 法	说　　明
JCheckBox()	创建一个没有文本、没有图标并且最初未被选定的复选框
JCheckBox(String text)	创建一个带指定文本的、最初未被选定的复选框
JCheckBox(String text, boolean selected)	创建一个带指定文本的复选框，并可以指定其最初是否被选择
JCheckBox(String text, Icon icon)	创建带有指定文本和图标的、最初未选定的复选框
JCheckBox(String text, Icon icon, boolean selected)	创建带有指定文本和图标的、最初未选定的复选框，并可以指定其最初是否被选择

JCheckBox的常用方法

方 法 原 型	说　　明
void setSelected(boolean b)	设定复选框的选择状态，true为被选择，false为不被选择
boolean isSelected()	返回复选框的选择状态
void setText(String text)	设置复选框的文本
String getText()	返回复选框的文本
void setIcon(Icon icon)	设置复选框的图标

javax.swing.JRadioButton

- JRadioButton提供单选按钮组件，其构造方法共有8种重载，以下是常用的几种：

构 造 方 法	说　　明
JRadionButton()	创建一个没有文本、没有图标并且最初未被选定的单选按钮
JRadionButton (String text)	创建一个带指定文本的、最初未被选定的单选按钮
JRadionButton (String text, boolean selected)	创建一个带指定文本的单选按钮，并可以指定其最初是否被选择
JRadionButton (String text, Icon icon)	创建带有指定文本和图标的、最初未选定的单选按钮
JRadionButton (String text, Icon icon, boolean selected)	创建带有指定文本和图标的、最初未选定的单选按钮，并可以指定其最初是否被选择

JRadioButton常用方法

方 法 原 型	说　　明
void setSelected(boolean b)	设定单选按钮的选择状态，true为被选择，false为不被选择
boolean isSelected()	返回单选按钮的选择状态
void setText(String text)	设置单选按钮的文本
String getText()	返回单选按钮的文本
void setIcon(Icon icon)	设置单选按钮的图标

javax.swing.ButtonGroup

- 事实上，单选按钮本身并不具备单选效果，它必须依靠按钮组才能达到单选的目的；
- ButtonGroup用来提供按钮组，将一系列按钮加入到同一个按钮组中，那么同一按钮组中的按钮只能有一个被选择；
- ButtonGroup的构造方法如下：

构 造 方 法	说　　明
ButtonGroup()	创建一个新的按钮组

ButtonGroup的常用方法

方 法 原 型	说　　明
void add(AbstractButton button)	将指定按钮添加到按钮组中
int getButtonCount()	返回按钮组中按钮的数量
void remove(AbstractButton button)	将指定按钮从按钮组中删除
Enumeration getElements()	返回按钮组中所有的按钮

javax.swing.JComboBox

- 使用JComboBox可以创建组合框组件，也就是俗称的下拉列表，以下是它的4种构造方法重载：

构 造 方 法	说　　明
JComboBox()	创建一个空的组合框
JComboBox(Object[] items)	使用数组中的元素作为选项的组合框
JComboBox(Vector items)	使用集合中的元素作为选项的组合框
JComboBox(ComboBoxModel model)	创建具有指定数据模型的组合框

JComboBox的常用方法

方 法 原 型	说　　明
void addItem(Object item)	为列表添加选项
void insertItemAt(Object item, int index)	在指定索引位置添加指定的选项
int getSelectedIndex()	返回被选择的选项的索引
Object getSelectedItem()	返回被选择的选项
void setSelectedIndex(int anIndex)	设置指定索引位置的选项被选择
void setSelectedItem(Object anItem)	设置指定的选项被选择
void removeItem(Object anItem)	删除指定的选项
void removeItemAt(int anIndex)	删除指定索引位置的选项
int getItemCount()	返回列表中所有选项的数量

表单组件示例

总结

- 可以使用JDialog来创建对话框，编写多窗口程序；
- 使用JScrollPane来为某些大视图的组件提供滚动面板；
- 文本组件是指专门用来操作文字的组件；
- 适当的使用表单组件可以使用户界面更加人性化，最大限度地方便用户操作。