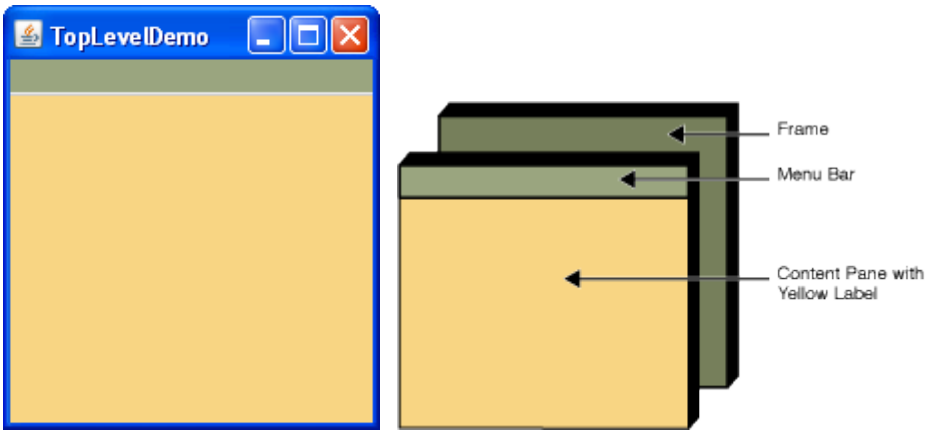
Java 图形界面

组件和容器？

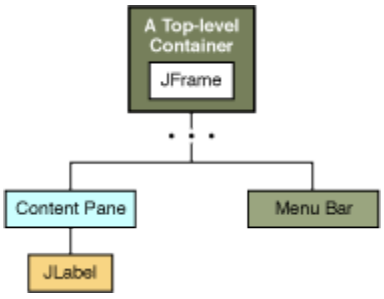
http://docs.oracle.com/javase/tutorial/uiswing/components/button.html

要掌握 Java 图形界面开发，首先需理解组件和容量两个概念。组件放置在容器中，容器包含组件。日常生活中，书包就是一个容器，而书包里的一本书、一支笔都是组件。当你把书包放到车里时，车就变成容器，而书包就成为组件。容器可以被放置到更大的容器中，因此容器可以包含组件，也可以包含其他的容器。容器之间的包含关系也反映容器之间层级关系，上一级的容器自然可以包含下一级的容器。组件如何放置到容器中？Swing 中的各种 Layout 负责解决。

Java 提供现成的用户界面程序开发工具包 Swing。开发人员只需调用 Swing 中的组件和容器，就可以完成界面开发，十分便利。前面已经提过容器的层级问题，JFrame 是 Swing 中常用的最高层级。图**是一 JFrame 的例子，Frame 内部放置一菜单栏和 contentPane，contentPane 中放置一黄色的标签。



上述各对象的层级关系如下：



注意 JLabel 没有直接放置，而是通过简接地放置在 Content Pane 实现放置在 Frame 中。具体的代码见文件夹 MyFrame。注意放置 JLabel 部分代码：

```
frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
```

add 方法中有 BorderLayout.CENTER 参数，就是设置 Label 在 ContentPane 中放置的方式与位置。当然还有很多其他的 Layout 方式，读者可以在应用时查阅官方 API 文档或者其他相关资料，此处不再赘述。

尝试在 Content Pane 中增加一个按钮组件，Swing 中的按钮是类 JButton，代码如下：

```
jb = new JButton("press me");
jb.setPreferredSize(new Dimension(50, 20));
```

按钮被放置到 content pane 的下方。

日常经验显示当按钮被点击后，能看到界面对应的反馈，比如弹出新的对话框。整个过程可由如下的链条描述：点击动作->触发事件->捕捉事件->进行对应的处理。当鼠标点击按钮时，产生按钮点击事件，如何区分不同按钮的点击事件？对按钮而言，可以给每个按钮设置一个身份标签，不同的按钮拥有自己独一无二的标签区分，通过方法 setActionCommand 实现。对按钮事件的处理，则通过 actionPerformed 方法。

小目标：以 MyFrame 中的按钮为例，假设要实现如下的功能，每点击一次按钮，按钮上显示当前的点击次数。

按钮上的文字要求能够更新，这种需求不是现在才有，以前的程序员也有，因此可确定 Java 肯定会提供对应的 API。打开 Java API 文档，定位到 JButton 类后，可在 Method Summary 下面看到 8 个方法，但没有一个符合我们的需求。再往下看，可发现 JButton 继承大量来自 javax.swing.AbstractButton，javax.swing.JComponent，java.awt.Container 和 java.awt.Component 的方法。即使只是从字面意思去揣测哪个是目标方法，面对这么多方法，人也有点不知所措。

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, bounds, checkImage, checkImage, coalesceEvents, contains, createImage, createImage, createVolatileImage, createVolatileImage, disableEvents, dispatchEvent, enable, enableEvents, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getBackground, getBounds, getColorModel, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeysEnabled, getFont, getForeground, getGraphicsConfiguration, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getLocale, getLocation, getLocationOnScreen, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hasFocus, inside, isBackgroundSet, isCursorSet, isDisplayable, isEnabled, isFocusable, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, processComponentEvent, processFocusEvent, processHierarchyBoundsEvent, processHierarchyEvent, processInputMethodEvent, processMouseWheelEvent, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setFocusable, setFocusTraversalKeysEnabled, setIgnoreRepaint, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle

此时应该停下来想一想，目标是在按钮上设置文字，自然和关键词 text 有关（为什么不是 String？这依赖于经验，一般是 text 优先；如果 text 不行，再考虑 String 也不迟。），因此可以基于关键词 text 进行查找。 AbstractButton 中有大量的 Text 相关的方法。因为是要设置按钮文字，所以凡是“get”前缀的方法，可以排除。

Methods inherited from class javax.swing.AbstractButton

actionPropertyChanged, addActionListener, addChangeListener, addImpl, addItemListener, checkHorizontalKey, checkVerticalKey, configurePropertiesFromAction, createActionListener, createActionPropertyChangeListener, createChangeListener, createItemListener, doClick, doClick, fireActionPerformed, fireItemStateChanged, fireStateChanged, getAction, getActionCommand, getActionListeners, getChangeListeners, getDisabledIcon, getDisabledSelectedIcon, getDisplayedMnemonicIndex, getHideActionText, getHorizontalAlignment, getHorizontalTextPosition, getIcon, getIconTextGap, getItemListeners, getLabel, getMargin, getMnemonic, getModel, getMultiClickThreshhold, getPressedIcon, getRolloverIcon, getRolloverSelectedIcon, getSelectedIcon, getSelectedObjects, getText, getUI, getVerticalAlignment, getVerticalTextPosition, imageUpdate, init, isBorderPainted, isContentAreaFilled, isFocusPainted, isRolloverEnabled, isSelected, paintBorder, removeActionListener, removeChangeListener, removeItemListener, setAction, setActionCommand, setBorderPainted, setContentAreaFilled, setDisabledIcon, setDisabledSelectedIcon, setDisplayedMnemonicIndex, setEnabled, setFocusPainted, setHideActionText, setHorizontalAlignment, setHorizontalTextPosition, setIcon, setIconTextGap, setLabel, setLayout, setMargin, setMnemonic, setMnemonic, setModel, setMultiClickThreshhold, setPressedIcon, setRolloverEnabled, setRolloverIcon, setRolloverSelectedIcon, setSelected, setSelectedIcon, setText, setUI, setVerticalAlignment, setVerticalTextPosition

对比剩下的方法，setText 的可能性最大。点击查看 setText 方法的说明，可确定就是我们需要的方法。

setText

```
public void setText(String text)
```

Sets the button's text.

Parameters:

text - the string used to set the text

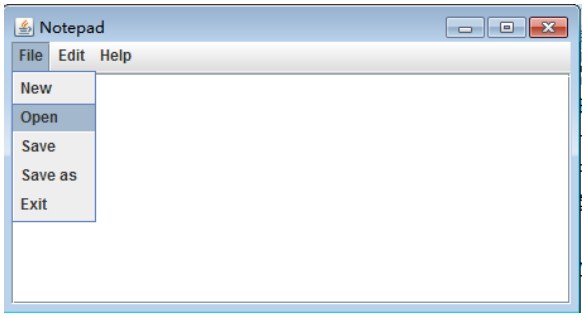
See Also:

getText()

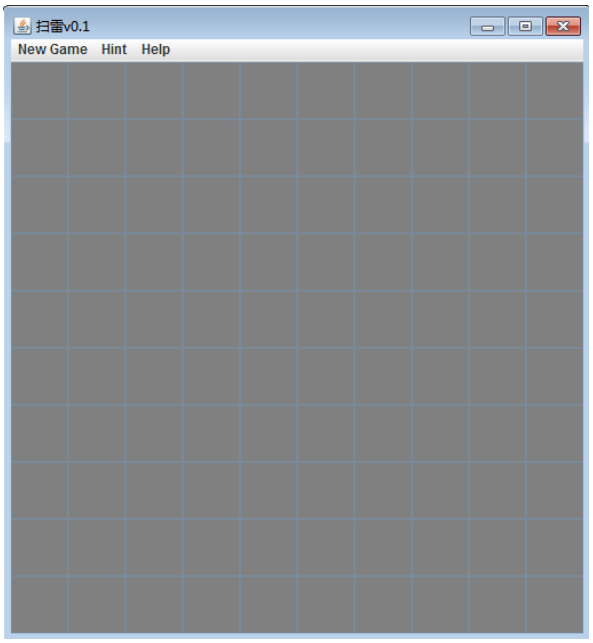
另外要注意的一点，actionPerformed 方法由谁来实现是个很重要的问题。根据面向对象设计中“自己的事情自己负责”原则，每个按钮应该有权利指定：发生在自身的点击事件由谁来处理。具体则由方法 addActionListener 实现（作者怎么知道要用该方法？因为作者看过示例代码，反推得出这样的结论呀!!），查阅该方法，发现一个输入参数是一个接口 ActionListener，其中有唯一的方法：void actionPerformed(ActionEvent e)。只要有一个实现 ActionListener 接口以及其中的 actionPerfromed 方法的类，再将该类与按钮绑定，就可以让发生在按钮上的事件由该类中的 actionPerfromed 方法来实现。

请读者参考官方 Java API 文档或者查找书籍，实现小目标；如果还不明白上述的文字，请查看官方文档中的示例代码。小目标的示例代码见文件夹 MyFrame2。其他组件的事件处理与按钮的处理类似，应用时请参照官方文档或者其他书籍即可。学以致用，我们来看两个 UI 程序界面的案例。

案例 1:
第一个是简单的记事本界面，具体代码见 Notepad 文件夹，使用到的组件有 JMenuBar，JMenu，JMenuItem，JScrolPane，JText 等。



案例 2:
如下的扫雷游戏的界面，主要是 JButton 的二维排列，并且每个按钮点击后改变颜色，具体代码见 MineView 文件夹。



Java 图形界面的设计开发介绍到此，主要的方法是参考示例代码和查阅官方 API 文档。实际的开发过程中，采用可视化的开发工具，比如 windowbuilder，将界面的开发过程转化为画图的过程，提高开发速率。