JavaSwingAPIs可插拔式外观风格特性应用 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/240/2021_2022_JavaSwingA_c103_240655.htm Java Swing 应用编程接口提供了可插拔式外观风格（PLAF)的特性，它允许Swing 图形用户接口（GUI）小应用根据编程人员定制的外观风格设置来改变界面外观风格。几乎所有现代用户接口框架都结合了视图和控制，无论它们是基于SmallTalk、C 或Java。 Swing将每个组件的视图和控制封装到一个称为UI委托的对象中。 因此，Swing的体系结构被称为模型委托结构而不是模式-视图-控制器结构。理想情况下，模型和UI委托直接的通讯是间接的，允许多个模型与一个UI委托相连，反之亦然。UI委托 每个UI委托源自一个名为ComponentUI的抽象类。ComponentUI的方法描述了一个UI委托和使用它的组件之间进行通讯的基本原理。 注意的是每个方法都将JComponent作为一个参数。ComponentUI有很多方法，这里只给出几个最重要的： static ComponentUI createUI(JComponent c)：该方法通常用来返回UI委托的一个共享实例，该UI委托通过定义ComponentUI子类本身而定义。这个共享实例用于相同类型的组件之间的共享（例如，所有使用金属外观的JButtons共享同样的静态UI委托实例，默认情况下，该委托实例在javax.swing.plaf.metal.MetalButtonUI中定义。 InstallUI(JComponent c)：该方法在特定的组件上安装ComponentUI。通常会给组件和它的模型添加一个监听器，当状态发生改变时来通知UI委托进行视图的更新。 Update (Graphics g, JComponent c)：如果组件是不透明的，那么应该描绘它的背景并调用paint（Graphics g,JComponent C）方法。

Paint (Graphics g, JComponent c):为了能够正确地描绘，该方法要从组件收集所有需要的信息以及可能的模型。 为了增强特定UI委托的使用，我们可以调用一个组件的setUI（）方法，如下所示：JButton m_button = new JButton().m_button.setUI((MyButtonUI)MyButtonUI.createUI(m_button)). JComponent类中定义了用于分配UI委托的方法，因为方法声明中不包含特定组件代码。然而，对数据模型而言这是不可能的，因为不存在所有模型可以追溯到的基接口（例如，不存在像Swing模型中的ComponentUI此类的基类）。为此，分配模型的方法在JComponent的子类中定义。使用PLAF Swing包含几个UI委托集。每个集合中包含了用于大部分Swing组件的ComponentUI实现，且每个这样的集合称为一个PLAF实现。 javax.swing.plaf包有继承自ComponentUI的抽象类组成，javax.swing.plaf.basic包中的类扩展了这些抽象类用来实现基本的外观。UI委托集合是所有其它外观类用作构建自己的外观的基类。基本外观不能自己使用因为BasicLookAndFeel是一个抽象类。有三个继承自BasicLookAndFeel的可插拔式外观实现： Windows: com.sun.java.swing.plaf.windows.WindowsLookAndFeel CDEMotif: com.sun.java.swing.plaf.motif.MotifLookAndFeel Metal (default): javax.swing.plaf.metal.MetalLookAndFeel 同时还有一个模仿苹果用户接口的MacLookAndFeel，但是没有包含在Java2中，必须单独下载。多路外观javax.swing.plaf.multi.MultiLookAndFeel扩展了javax.swing.plaf中的所有抽象类。它运行同时使用多种外观的组合，且有意但并不仅限于和访问外观一起使用。每个多路UI委托的任务

是管理每个它们的子UI委托。 每个外观包中都包含了一个继承自抽象类javax.swing.LookAndFeel：BasicLookAndFeel, MetalLookAndFeel, WindowsLookAndFeel等的类。这是访问每个外观包的中心点。当改变当前外观时，你会用到它们。同时UIManager类（用于管理安装的外观）使用它们来访问当前外观UIDefaults表（其中包含了用于每个Swing组件相应外观的UI委托类名称）。 要想改变应用的当前外观，你只需要简单调用UIManager的setLookAndFeel（）方法，并将要使用的外观全名传递给该方法即可。可以使用列表A中的代码在运行时完成该任务。列表A try { UIManager.setLookAndFeel("com.sun.java.swing.plaf.motif.MotifLookAndFeel"). SwingUtilities.0updateComponentTreeUI(myJFrame). } catch (Exception e) { System.err.println("Could not load LookAndFeel"). } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com