JavaGUI三剑客风云争霸 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/237/2021_2022_JavaGUI_E4_B8_89_c104_237340.htm 进行Java GUI（Graphical User Interface，图形用户界面）编程，大家或许经常徘徊在SWT/JFACE, Swing, AWT之间选择，哪一个更合适自己？AWT作为Java语言的第一个GUI类库包，在这三者之间中年龄最长，可谓开国元勋；老二Swing，兼容AWT，同时又对AWT进行了改进，可谓站在前辈的肩膀上，自然就会看的远了；老三SWT/JFace，则只能用不走寻常路来形容它，SWT/JFace采取了与AWT和Swing完全不同的技术路线。这三剑客之间，究竟孰优孰劣，且听下文分解。 1. 穷途末路的AWT AWT(Abstract Windowing Toolkit)，中文译为抽象窗口工具包，是Java提供的用来建立和设置Java的图形用户界面的基本工具。AWT由Java中的java.awt包提供，里面包含了许多可用来建立与平台无关的图形用户界面(GUI)的类，这些类被称为组件(components)。 抽象窗口工具包 （Abstract Windowing Toolkit） (AWT)是Java的平台独立的窗口系统， 图形和用户界面器件工具包。AWT是Java基础类 (JFC)的一部分，为Java程序提供图形用户界面(GUI)的标准API。 AWT提供了Java Applet和Java Application中可用的用户图形界面GUI中的基本组件(components)。由于Java是一种独立于平的程序设计语言，但GUI却往往是依赖于特定平台的，Java采用了相应的技术使得AWT能提供给应用程序独立于机器平台的接口，这保证了同一程序的GUI在不同机器上运行具有类似的外观（不一定完全一致）。 抽象窗口工具包AWT (Abstract Window

Toolkit) 是 API为Java 程序提供的建立图形用户界面GUI (Graphics User Interface)工具集，AWT可用于Java的applet和applications中。它支持图形用户界面编程的功能包括： 用户界面组件；事件处理模型；图形和图像工具，包括形状、颜色和字体类；布局管理器，可以进行灵活的窗口布局而与特定窗口的尺寸和屏幕分辨率无关；数据传送类，可以通过本地平台的剪贴板来进行剪切和粘贴。 然而，Java推出的时候，AWT作为Java最弱的组件受到不小的批评。最根本的缺点是AWT在原生的用户界面之上仅提供了一个非常薄的抽象层。例如，生成一个AWT的复选框会导致AWT直接调用下层原生例程来生成一个复选框。不幸的是，一个Windows平台上的复选框同MacOS平台或者各种UNIX风格平台上的复选框并不是那么相同。这种糟糕的设计选择使得那些拥护Java “ 一次编写，到处运行 （write once, run anywhere） ” 信条的程序员们过得并不舒畅，因为AWT并不能保证他们的应用在各种平台上表现得有多相似。一个AWT应用可能在Windows上表现很好，可是到了Macintosh上几乎不能使用，或者正好相反。在90年代，程序员中流传着一个笑话:Java的真正信条是 “ 一次编写，到处测试 （write once, test everywhere） ” 。导致这种糟糕局面的一个可能原因据说是AWT从概念产生到完成实现只用了一个月。 在第二版的Java开发包中，AWT的器件很大程度上被Swing工具包替代。Swing通过自己绘制器件而避免了AWT的种种弊端。Swing调用本地图形子系统中的底层例程，而不是依赖操作系统的高层用户界面模块。Swing的出现，宣告了AWT的穷途末路，目前几乎看不到AWT在GUI上的应用了。2. Swing想说爱你不容易 Java Swing是Java Foundation

Classes（JFC）的一部分，它是试图解决AWT缺点的一个尝试。从这一点上来说，Swing可以说是站在前人（以AWT的表现，实在很难称之为巨人）的肩膀上了。SWING解决了AWT的很多缺点。相对于AWT, Swing是轻量级元件。SWING 提供许多比AWT更好的屏幕显示元素。它们用纯Java写成，所以同Java本身一样可以跨平台运行，这一点不像AWT。 它们是JFC的一部分。 它们支持可更换的观感和主题（各种操作系统默认的特有主题），然而Swing不是真的使用原生平台提供的设备，而是仅仅在表面上模仿它们。这意味着你可以在任意平台上使用JAVA支持的任意观感。 轻量级元件的缺点则是执行速度较慢，优点就是可以在所有平台上采用统一的行为。 在Swing 中，Sun 开发了一个经过仔细设计的、灵活而强大的 GUI 工具包。其中大量应用了MVC模式，这大大增加了Swing的灵活性。笔者曾经做过一个大型的大型C/M/S（Client/Middleware/Server）项目，其中客户端UI采用的就是Swing，可以说，Swing几乎可以实现所有的你能够想到效果，只要你技术足够精湛，都可以实现。这也许在某些高手看来，是Swing一个很明显的优势。然而Swing的这种设计确苦了Java的初学者或者面向对象程序设计造诣不深的程序员。灵活就意味这功能强大，功能强大就意味着复杂，对于一般的程序员来说，Swing太复杂了，以至于他们在还不了解Swing的时候就已经放弃了选择Swing，或者失去净下心来继续学下去的毅力，最后写出来的只能是一堆垃圾代码。 如果说功能强大但是过于复杂会让人对Swing想爱确不知道怎么去爱的话，那么Swing的低效则会让大多数的程序员感叹Swing，想说爱你不容易。由于Swing是轻量级组件，因

此Swing中的每一个组件都是采用Java身的画点、画线的函数画出来的，并没有调用操作系统组件。Java字节码的运行速度大概是同等条件下C/C 语言程序运行速度的1/10～1/5。于是，采用JBuilder进行开发的朋友们经常可以看到JBuilder灰屏（窗体上组件还没有画出来）的景象。正是因为Swing的蜗牛速度，因此在Java推出这么多年来，很少能够看见比较成熟的Swing桌面应用（JBuilder算是其中最成功的一个了，但是现在随着Eclipse的崛起，JBuilder的发展也是举步维艰）。 总之，Swing在AWT的基础上很好的解决了跨平台观感不一的问题，并且提供了比AWT更为丰富的组件（AWT连树形控件、表格控件都没有）和强大的功能，却因为其过于复杂难以上手和让人无法接受的速度让广大程序员对其失去了好感。这不能不让人惋惜。 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