Java秘史：隐藏在SWT_Swing背后的故事 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022_Java_E7_A7_98_E5_8F_B2_c104_252392.htm 要想弄清楚为什么一切都被弄得如此混乱，要从几年前只存在AWT的时候说起。SUN当时已经建立了一套基本的可移植控件类，这些类映射到不同操作系统上的原生窗口组件（native widget），显然下一步应该继续增强这套模型，除了初始的CUA 92组件（文字、按钮等等），再继续加上表格、树、记事本、滑块等等……当时的AWT还满是漏洞，远不能称为可靠，还需要SUN的coder们去修补。SUN的developer们如Graham和Otto总是习惯于公开把他们的bug归咎为操作系统的差异，比如“ Windows和OS/2的焦点次序不同 ”或者“ 在……之间Ctrl-X的行为不一样 ”，以及其他苍白的托辞，好让批评的火力从SUN太早释出代码这个问题的真相上移开。然后Amy Fowler来到了SUN。不是我大男子主义，Amy是个聪明的美女，大多数呆头呆脑只懂技术的开发人员都要被她捏在手里。 Amy来自一家Smalltalk公司，叫做Objectshare，在那里她负责搞UI类库。 跟Java相比Smalltalk的历史有些悲惨，曾几何时有3家庞大的Smalltalk公司IBM、Parc-Place和Digitalk。在90年代初期3家公司的市场份额大致相等，生活是美好的。Parc-Place采用仿窗口部件（emulated widgets）的设计（即Swing的设计），IBM和Digitalk则采用原生窗口部件（native widgets）。后来IBM压倒了另外两家，因此他们打算合并成一家，假设叫做Parc-Place Digitalk。随后当他们试图将他们的产品融合到一个叫做Jigsaw的计划中时爆发了一场大战，计划由于政治原因

失败了（开发人员实际上已经能让它运转起来），就因为原生和仿造两派的死战。 Amy赢得了精神上的胜利，不过在IBM我们赢得了他们所有的生意，因为这两家公司在一整年里除了吵架什么都没做。当尘埃落定之后PPD（Parc-Place Digitalk当时已改名为Objectshare，跟Windscale改名为Sellafield的原因相同让人们淡忘之前发生的灾难）的股票价格从60美元掉到了低于1美元1股。他们因为伪报收入被NASDAQ摘牌，从此消失。此时SUN正走上与PPD类似的技术方向，于是PDD的技术人员都把他们的简历投到了SUN。Amy被雇佣了，她承诺通过轻量级方案解决所有窗口组件的问题，因此说服SUN管理层让她当了GUI开发部门的头头。她是拿着“这里原来的人都搞砸了，我是来解决的”的钥匙进来的。随后Amy雇佣了所有她过去在Parc-Place的旧朋友，让他们来开发Swing。 显然Swing应该做的是仅仅成为一个绘制框架，给那些希望创建地图软件或者绘图软件的人们使用，无论如何，应该围绕AWT类库来建造它，按钮之类的东西仍然交给AWT来管。SUN的人比如Philip和Mark已经让AWT能够处理表格、树和记事本（notebook，？），所以Swing的方向应该说很明显了。但那些毁了PDD的人不干，他们非要把一切都弄成轻量级的。由于SUN管理层的无知，再加上Amy无情的政治手段，造成了我们今天所见的混乱局面。Amy还使SUN相信Swing是作为Mozilla项目的一部分与Netscape联合开发的，事实上这只是她的宣传伎俩。 在IBM，我们从第一天起就憎恶Swing。庞大、满是错误，而且难看至极。原先我们的工具如VisualAge for Java都是用Smalltalk（用的是原生窗口组件）写的，所以当我们将这些工具向Java代码库迁移时，

我们需要一套窗口组件。IBM这边的开发人员都是原来搞Smalltalk的那一批人，我们对管理层要求用Swing来构建WebSphere Studio工具都非常不情愿。Swing是个可怕的充满缺陷的怪兽。 在WebSphere Studio最初的预览中，当与Microsoft Visual Studio作对比演示的时候，我们所有的客户都讨厌它，就因为它的外观，而不管它的功能有多强。大多数消费者都不会买一辆让人觉得难看的车，哪怕这车有一台出色的引擎。因此我们开始了一个项目，是把我们的Smalltalk原生窗口组件移植到Java上去。这个项目是加拿大的Object Technology International小组做的。这个项目获得了成功，被运用在在我们发布的VisualAge Micro Edition产品中，VisualAge Micro Edition后来成为J2ME开发方面一个非常成功的IDE。但是OTI的人发现，Swing在读取Windows事件方面有极严重的缺陷，我们甚至无法进行SWT（S开始是Simple的缩写，不过后来变成了Standard的缩写）和Swing间的互操作。他们在读事件队列的时候用了一种可能留下内存漏洞的方式，所以我们不得不采用我们自己的查询Windows事件队列的循环，以纠正这个错误。 我们试了一次又一次让SUN修复这个错误，但Amy就是听不进去，所以我们才决定SWT和AWT/Swing不能共存。我们甚至在SWT中定义了自己的Point和Rectangle类整个工具包对AWT或Swing都没有任何依赖。我们把这个工具包放到了Eclipse中，这是一个工具平台，它的总体设计目标就是要战胜Micrsoft和Visual Studio。Eclipse是开源的，所以任何人都可以在上面构建自己的东西，我们已经有像TogetherSoft和Rational这样的公司移植到了上面。我们的竞争者是Microsoft，所以我们所有努力和注意

力都是从正面针对Microsoft。 不管怎么说SUN对此非常不满。他们的Netbeans跟Eclipse做的是相同的事，因此他们向IBM高层抱怨。他们认为SWT是要将你绑到Windows上，这纯粹是胡说，因为SWT能通过GTK在Mac/Linux上运行，以及一大堆嵌入式平台。他们拒绝让Eclipse获得Java认证，因为里面有原生代码，所以Eclipse产品必须很小心地使用单词“ Java ”这个SUN的商标。Eclipse甚至不能把自己称为一个Java IDE，SUN已经威胁过要采取法律行动来制止IBM在任何时候把Eclipse称作一个Java IDE。结果之一就是IBM在Eclipse上创建的GUI设计工具，允许你构建Swing/AWT GUI，却不让你往里面拖放SWT窗口控件。 将SWT从Eclipse中分离出来是完全可能的，只需要把DLL抠出来放到路径中，并使用窗口组件工具包来给你的银行或者保险或者其他什么应用程序开发GUI。再次说明，我们无法更进一步，因为SUN把我们的双手绑上了。虽然作为Eclipse开放源码协议的一部分，CPL允许我们提供这样的解决方案，但SUN已经很清楚地表明他们不希望我们这样做。 对于用户社区来说，无论IBM和SUN的最终动机是什么，我发现有一点总是很有趣：喜爱Swing的人总会说“ 一旦你花上几年时间去掌握它，你就能正确地使用它 ”，这基本上是他们在试图证明和维护他们辛苦得来的用途有限的专门技术；而SWT的拥护者们说的是“ 哇，这真快，这跟原生的一样，还可以用XP皮肤……它还又轻又小 ”。有一句话是我喜欢的，我们的一个用户说，Swing就像Java决定不通过操作系统来实现原生的IO，而是通过磁头马达API自己来读磁盘的扇区。Swing基本上就是这样的，它拿着个底层的“ paint(Graphics) ”方法，自己来绘制所有的窗口组件。