基于Java的界面布局DSL的设计与实现 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/267/2021_2022__E5_9F_BA_E4_BA_8EJava_c104_267213.htm 界面设计应该是一项充满创造性、富有乐趣的工作，但是却往往被认为非常的枯燥和繁琐。究其原因，是因为界面布局领域所采用的描述概念和具体的实现语言之间存在很大的语义隔阂。而一般的界面开发工具提供的所见即所得以及界面布局管理器等方案也无法很好地解决这个问题。 在本文中，我们会给出一种更好的解决方案，我们不是去试图把界面设计者头脑中的设计概念和样式逐步降级、分解成所使用的实现语言能够理解的低层概念，也不是提供一些已经完成的、确定的但难以扩充和更改的布局样式库供界面设计者使用。我们所提供的是一种专门用于描述高层界面设计样式的语言。通过这种语言，界面设计者可以直接、明确地描述出他们头脑中的布局设计样式；通过这种语言，界面设计者可以自己方便地、灵活地制定自己需要的布局样式。此外，本文中给出的设计思想对于其他领域的设计也有很好的借鉴作用。 创造性，还是乏味？ 界面设计是一项非常有创造性，甚至富有艺术性的工作，一个简洁、易用、漂亮的界面在带给使用者方便的同时，也会给界面设计者带来极大的成就感。但是，在现实中，情况似乎并非如此，很多人都认为做界面是一项非常繁琐、机械、乏味的工作，并千方百计地去逃避界面相关的工作。这是为什么呢？原因很简单，因为做界面其实涉及两项工作，一项是界面的一些设计创意，包括界面的布局样式以及和使用者的交互方式，这项工作充满挑战和乐趣。但是，这些设计创意最终是

要落实到实现上的，这就是第二项工作。此时，你头脑中那些清晰、完整的设计概念开始变得琐碎，你不得不和那些低层次的坐标位置打交道。更糟糕的是，当你好不容易做好了一个界面，但是发现其中某些元素的布局需要一些调整时，这个你本应认为是一个很简单的改变却造成大量重复的低层次坐标位置更改时，你肯定会认为做界面是多么的机械和乏味呀！其实，造成这种认识的根源在于界面设计创意和实现这些创意概念的语言之间存在很大的断层。这样，在具体实现时，你就必须得把这些清晰、完整的布局样式降级成一些琐碎、没有什么意义的低层次的坐标值，使得实现语言能够理解。这项工作不仅乏味，而且最终的实现也非常的脆弱 一个在布局样式层面非常简单的更改，就会造成实现层面的巨大变动。比如：我们可以说把一组元素同时按比例缩小 10%，做过界面的朋友肯定知道这个更改意味着什么。 为了应对这个断层的问题，目前几乎所有的涉及界面制作的开发工具都提供了相同的解决方法：可视化的界面设计工具以及布局管理器。但是这两种方法都没有从根本上解决这个问题。 可视化界面设计工具确实避免了不少繁琐的界面元素摆放工作，但是对于专业的界面设计来说，通过拖放设计出来的界面在准确度和规范性上都有待提高，此外还有更为重要的一点，那就是存在于设计者头脑中的布局样式仍然没有被明确地描述出来，而是被降级成一个个摆放在一起的零散的组件，虽然这些组件本身是可视的。这个语义断层的存在同样会使得通过可视化界面设计工具设计出来的界面非常脆弱。 布局管理器试图通过提供一些常用的布局样式来解决这个问题。但是，这种做法非常僵化，也就是说你只能使用现有的布局

管理器，如果它们无法满足你的要求，你也无法自己定制。此外，这些布局管理器仅仅适合于一些简单的情况。对于一些复杂的布局样式来说，它们的描述能力就显得非常的不足。那些曾经和 GridBagLayOut 斗争过的朋友对此肯定深有体会。 在本文中，我们会给出一种更好的解决方案，我们不是去试图把界面设计者头脑中的设计概念和样式逐步降级、分解成所使用的实现语言能够理解的低层概念，也不是提供一些已经完成的、确定的但难以扩充和更改的布局样式库供界面设计者使用。我们所提供的是一种专门用于描述高层界面设计样式的语言。通过这种语言，界面设计者可以直接、明确地描述出他们头脑中的布局设计样式，通过这种语言，界面设计者可以自己方便地、灵活地制定自己需要的布局样式。也就是说，本来仅存在于界面设计者头脑中的抽象布局样式，现在也变得可描述，可编程了。 界面布局语言介绍 在学习界面布局语言的设计之前，先来了解一下该语言的使用是非常有帮助的。我们的界面布局语言非常简单，简单到只有一种原子：Component。Component 是一种基本的布局元素，可以对 Component 进行平移和伸缩，使其和给定的一个布局空间 Rectangle 匹配。比如对于 Button 这个 Component 来讲，它具有传统按钮的外观，但是它在布局上所占的实际空间则是由为其指定的 Rectangle 决定的。此外，Component 要最终在界面上显示出来，就必须有一个物理上的 Container。也就是说，只要给定了一个 Rectangle 和一个 Container，一个 Component 就可以在界面上指定的布局位置呈现出来。 例如，当我们使用布局语言在一个 JFrame 上坐标位置为 (0,0) 展示一个 width 为 200，height 为 60 的按钮时，我们可以这样来描

述（为了简洁起见，后面的代码实例中均略去 Layout 名字空间前缀）： Button().title(“ button1 ”).at(0,0,200,60).in(this.getContentPane()). 仅仅提供这样一种原子元素的语言显然无法满足我们前面提到的目标。在我们的界面布局语言中，还提供了两种在布局中非常常用的两种从已有组件构造新组件的组合手段：above 和 beside。其中 above 组合子接收 3 个参数：两个现有 Component 以及一个比例，它会产生出一个新的复合 Component，其中按照给定的比例把第一个 Component 摆放在第二个 Component 之上。Beside 组合子接收同样的 3 个参数，并且也产生出一个新的复合 Component，其中按照给定的比例把第一个 Component 摆放在第二个Component左边。 例如，如果我们希望在一个给定的 Container C 上的 Rectangle(0,0,300,40) 中，平行摆放一个 TextField 和一个 Button，且希望 TextField 占据 80% 的比例时，可以这样来描述： beside(TextField(), Button().title(“ ok ”), 0.8).at(0,0,300,40).in(C) 同样，我们可以使用 above 来进行如下描述： above(TextField(), Button().title(“ ok ”), 0.5).at(0,0,300,60).in(C) 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