##**11.3** 命名规则的力量

游戏程序员会抵制标准和约定 —— 并且有很好的理由：有些标准和约定非常刻板并且低效 —— 它们会毁坏创造性和程序质量。这真让人感到遗憾，因为有效的标准是你所能掌握的最强大的工具之一。本节将讲述为什么、何时以及如何创建自己的变量命名标准。

###为什么要有规则

命名规则可以带来以下好处。

- 要求你更多地按规矩行事。通过做一项全局决策而不是做许多局部决策，你可以集中精力关注代码更重要的特征。
- 有助于在项目之间的传递知识。名字的相似性能让你更容易、更自信地理解那些不熟悉变量原本应该是做什么的。
- 有助于你在新项目中更快速地学习代码。你无须了解 Anita 写的代码是这样的，Julia 是那样的，以及 Kristin 的代码又是另一种样子，而只须面对一组更加一致的代码。
- 有助于减少名字增生。在没有命名规则的情况下，会很容易地给同一个对象起两个不同的名字。例如，你可能会把总点数既称为 pointTotal，也称为 totalPoints。在写代码的时候这可能并不会让你感到迷惑，但是它却会让一位日后阅读这段代码的新程序员感到极其困惑。
- 弥补编程语言的不足之处。你可以用规则来仿效具名常量和枚举类型。规则可以根据局部数据、类数据语句全局数据的不同有所差别，并且可以包含编译器不直接提供的类型信息。
- 强调相关变量之间的关系。如果你使用对象，则编译器会自动照料它们。如果你用的编译语言不支持对象，你可以用命名规则来补充。诸如 address、phone 以及 name 这样的名字并不能表明这些变量是否想过。但是假设你决定所有的员工数据变量都应该以 Employee 作为前缀，则employeeAddress、employeePhone 和 employeeName 就会毫无疑问地表明这些变量是彼此相关的。编程的命名规则可以对你所用的编程语言的不足之处做出弥补。

关键之处在于，采用任何一项规则都要好于没有规则。规则可能是武断的。命名规则的威力并非来源于你所采用的某个特定规则，二三来源于以下事实：规则的村子为你的代码增加了结果，减少了你需要考虑的事情。

###何时采用命名规则

没有金科玉律表明何时一个建立命名规则，但是在下列情况下规则是很有价值的。

- 当多个程序员合作开发一个项目时
- 当你计划吧一个程序转交给另一个程序员来修改和维护的时候（这几乎总是会发生）
- 当你所在组织中的其他程序员评估你写到程序的时候
- 当你写的程序规模太大，以致于你无法再脑海里同时了解事情的全貌，而必须分而治之的时候
- 当你写的程序生命周期足够长，倡导你肯会在把它搁置几个星期或几个月之后又程序启动有关该程序的工作时
- 当一个项目存在一些不寻常的术语，并且你希望在编写代码阶段使用标准的术语或者缩写的时候

你一定会因使用了某种命名规则而受益。上述诸多注意事项将会帮助你决定在一个特定项目中按照何种程度来制定规则里所使用的规则的范围。

###正式程度

不同规则所要求的正式程度也有所不同。一个非正式的规则可能会像“使用有意义的名字”这样简单。下一节将会讲述其他非正式规则。通常，你所需的正式程度取决于为同一程序而工作的人员数量、程序的规模，以及程序预期的生命期。对于微小的、用完即弃的项目而言，实施严格的规则可能就太没有必要了。对于多人协作的大型项目而言，无论是在开始阶段还是贯穿整个程序的生命周期，正式规则但是成为提高可读性的必不可少的辅助手段。