[JUNIT]通过测试分类实现敏捷构建 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/144/2021_2022__5BJUNIT_5D_E9_80_9A_c104_144681.htm 人人都认可开发人员测试的重要性，但为什么运行测试还是需要花费太多时间？本月，Andrew Glover 揭示了三种用来确保端到端系统健壮性的测试类型，随后展示了如何按类型来自动排序及运行测试。即使是使用当今大型测试套件，这样做也能显著地减少构建时间。 如果这样说不会（令您）很痛苦的话，请设想您是一名任职于一家 2002 年早期创建的公司的开发人员。在金钱的驱动下，您和您的团队接到了一项任务，即使用最新且最强大的 Java&#8482. API 构建一个大型的数据驱动的 Web 应用程序。您和公司管理层都坚定不疑地相信这就是最终将被称为敏捷过程 的东西。从第一天起，您就用 JUnit 构建测试，且把它作为 Ant 构建过程的一部分尽可能频繁地运行。还将设置一个定时任务在夜间运行构建。在接下来的某个时刻，有人会下载 CruiseControl，不断增长的测试套件会在每次签入时运行。时至今日 经过过去几年的努力，您的公司已经开发了一个庞大的代码库和一个同样庞大的 JUnit 测试套件。一切都很正常，直到大约一年前，测试套件包含了 2000 个测试，同时人们开始注意到运行构建过程用时超过三个小时。在此之前的几个月，由于 CI 服务器资源紧张，您在代码签入时通过 Continuous Integration（CI）停止运行单元测试，并将测试切换到夜间运行，这使得之后的早晨时间非常紧张，于是开发人员努力去弄清楚是什么出错以及为什么出错。 这些天，似乎测试套件整晚极少超过一次运行，为什么会这样呢？因为

它们费时太多！没人会仅仅为了弄明白系统是否运行良好而几个小时守在那里。此外，整个测试套件都是在晚上运行，不是吗？ 由于测试运行得太不频繁，它们常常充满了错误。因而，您和您的团队开始质疑单元测试的价值：如果它们对代码质量那么重要，那又为什么会让人这么头痛呢？你们的结论是：单元测试有其重要的作用，但必须要能用一种更为敏捷的方式运行它们。尝试测试分类 您所需要的是一个将构建转换到一种更为敏捷状态的策略。您需要这样一种解决方案，使一天当中运行测试的次数超过一次，并使测试套件恢复到要用三个小时才能完成构建之前的水平。 为完整地恢复整个测试套件，在试图提出一个策略之前，很有必要弄清楚通用术语“ 单元测试 ”的含义。诸如“ 我家有一个动物 ”和“ 我喜欢车 ”这样的表述并不很具体，“ 我们编写单元测试 ”也是一样。这年头，单元测试能代表一切。 就拿之前有关动物和车的表述来说：它们导致了更多的疑问。例如，您家有哪种动物？是一只猫、一条蜥蜴还是一头熊？“ 我家有一头熊 ”和“ 我家有一只猫 ”截然不同。同样，当和汽车销售员交谈时，只说“ 我喜欢车 ”没什么用处。您喜欢哪种车：赛车、卡车还是旅行车？任何一个答案都能带来截然不同的结果。 同样，对于开发人员测试来说，按照类型 将测试分类也是很有用的。这样做能够实现更为精确的语言，并且能使您的团队以不同的频率运行不同的测试类型。为了避免运行所有“ 单元测试 ”所需的令人恐惧的三小时构建时间，分类是关键。 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