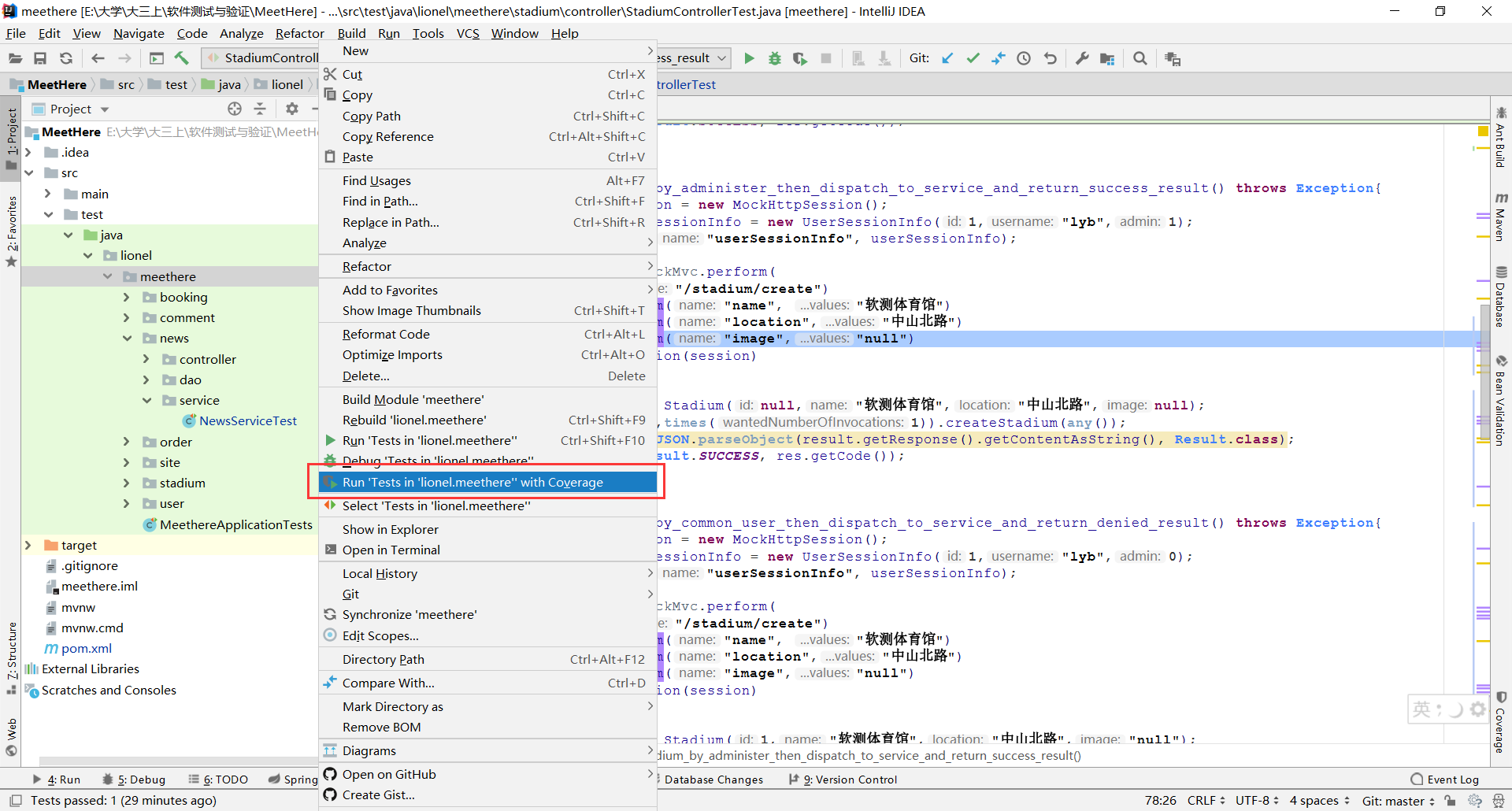
**覆盖度报告**

卢宇博

1. **代码覆盖度查看方法**

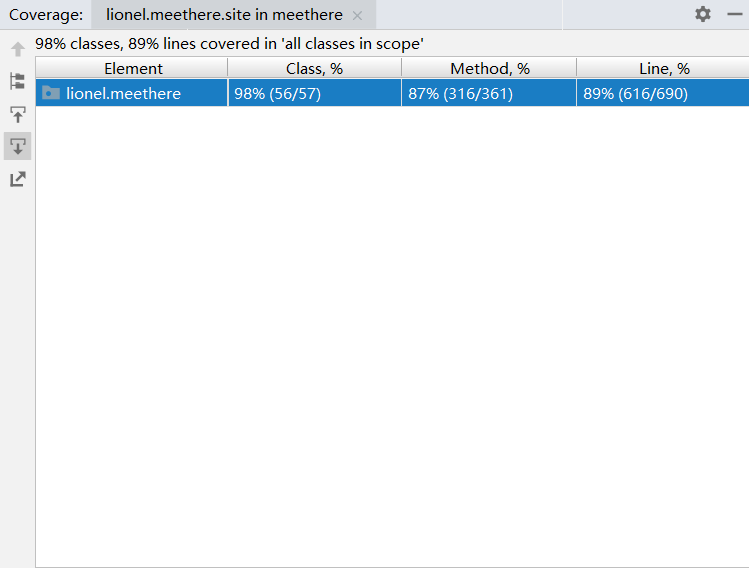
在进行单元测试时我们查看代码覆盖率的方法是利用Intellij IDEA直接查看



**代码覆盖度查看方法**

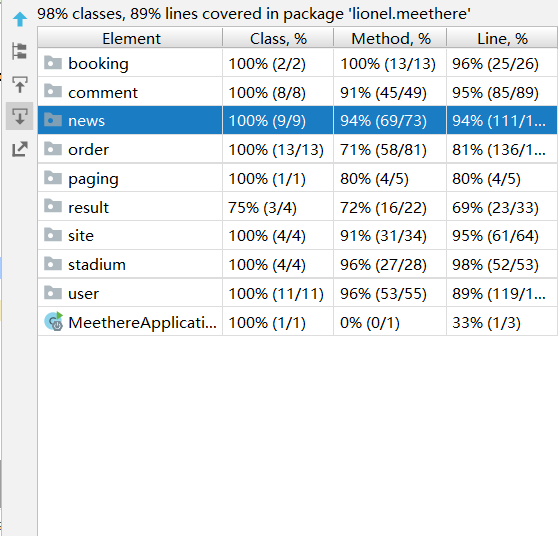
如图所示在文件夹上点击Run Tests with Coverage便可以在右边的框里看到Coverage 的结果，包括每个类的覆盖率。

1. **后端代码覆盖度分析**



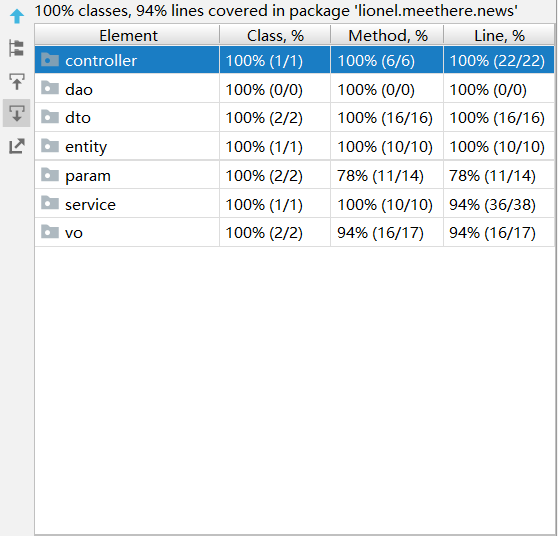
**整体后端代码覆盖度**

可以看出整个meethere项目的类代码覆盖度达到98%，方法代码覆盖度达到87%，代码行覆盖度达到89%



**具体类的代码覆盖度**

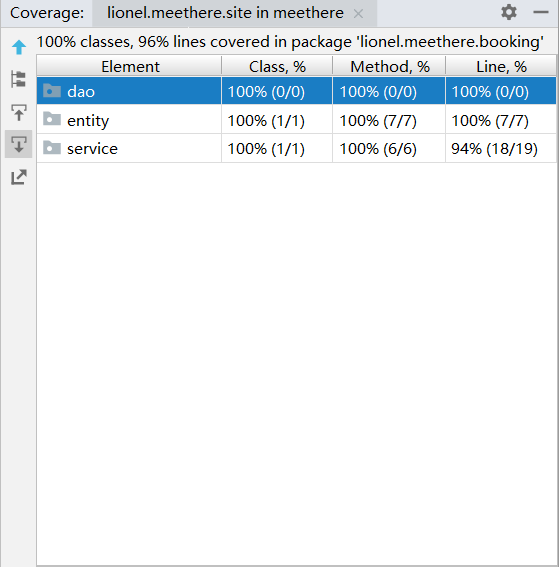
抛去页面类（paging），结果类（result）两个辅助实现的类，我们的大多数类的方法代码覆盖度和代码行覆盖度都比较高（为了实现业务层之间的数据传递，我们在order类中引入了SiteBookingOrderAdminVO与SiteBookingOrderUserVO两个value object对象，导致order类的代码覆盖度较低，同理其他类都有一些VO,DTO类降低了代码覆盖度）。



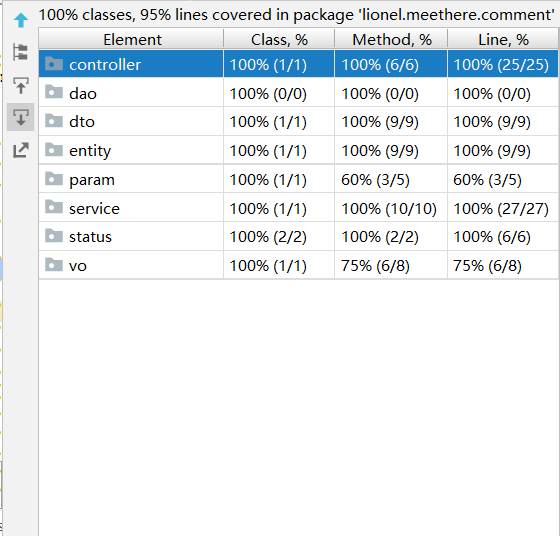
**news类的代码覆盖度**

以news类为例，可以看出我们的核心代码部分：主要做数据持久层的工作的dao层，主要负责业务模块的应用逻辑应用设计的service层，负责具体的业务模块流程的控制的controller层代码覆盖度都较高，反映了我们单元测试比较完整。

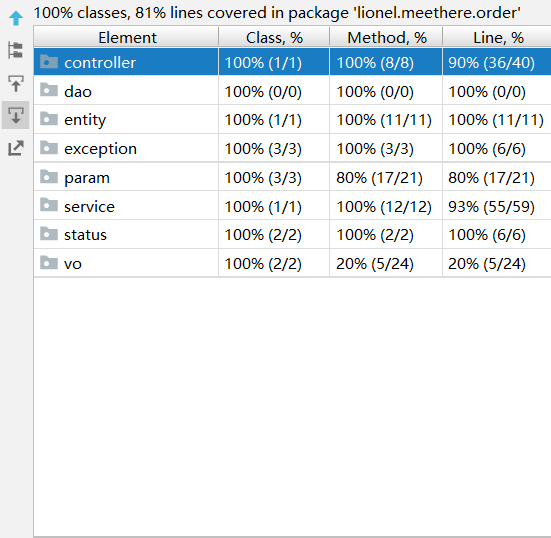
以下是其他主要类的代码覆盖度截图



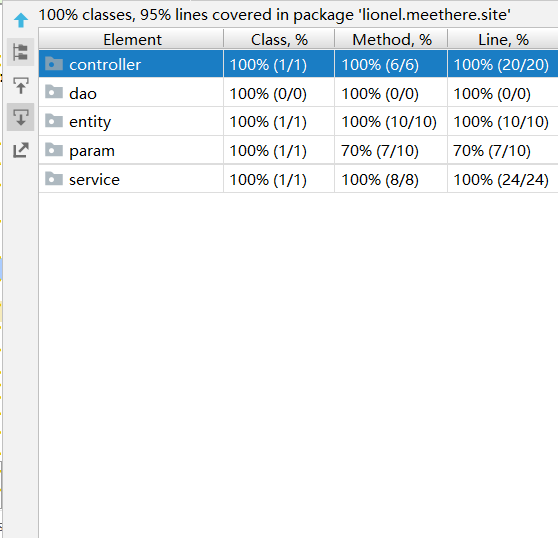
**booking类的代码覆盖度**



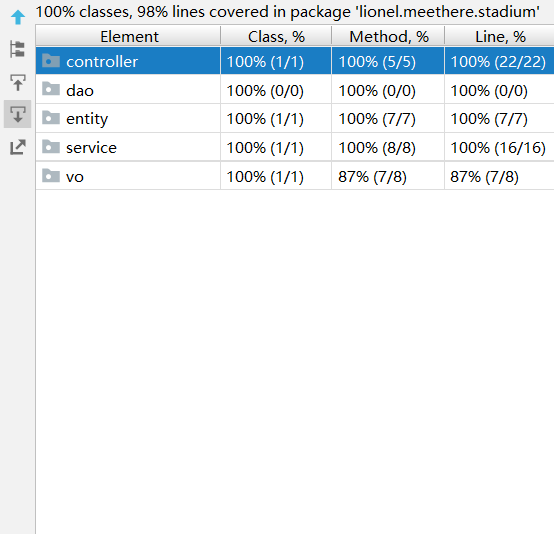
**comment类的代码覆盖度**



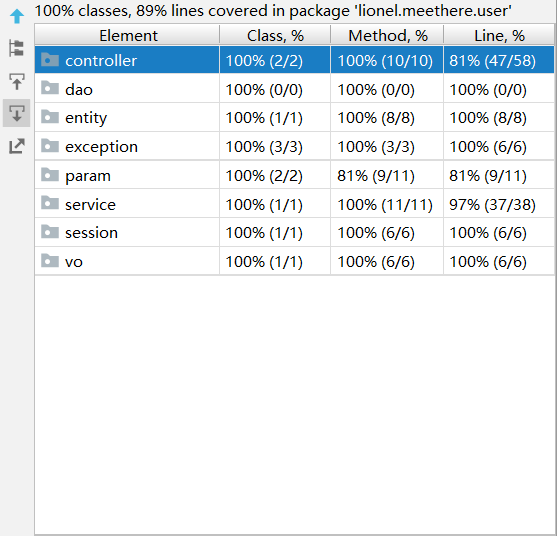
**order类的代码覆盖度**



**site类的代码覆盖度**

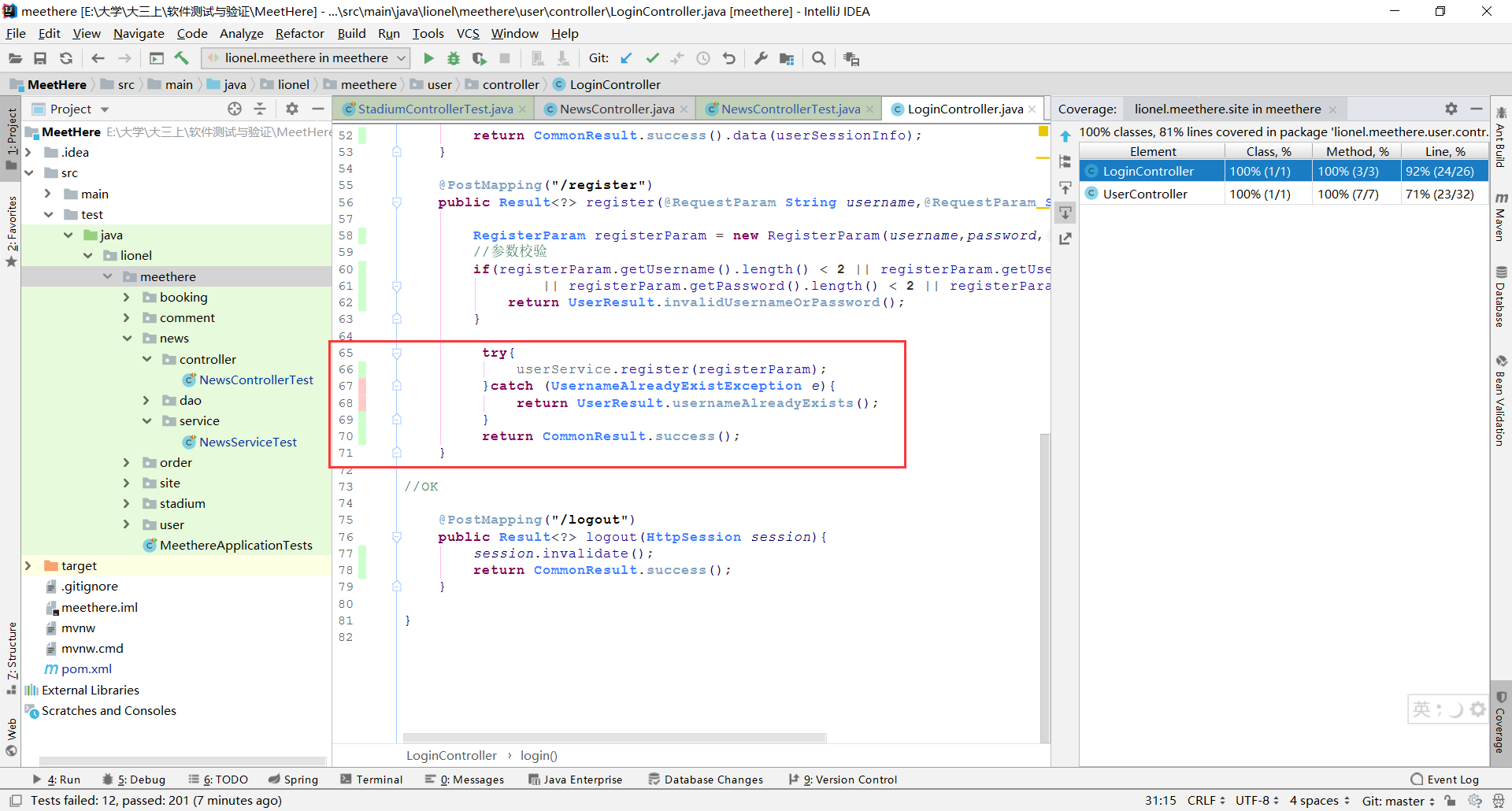


**stadium类的代码覆盖度**



**user类的代码覆盖度**

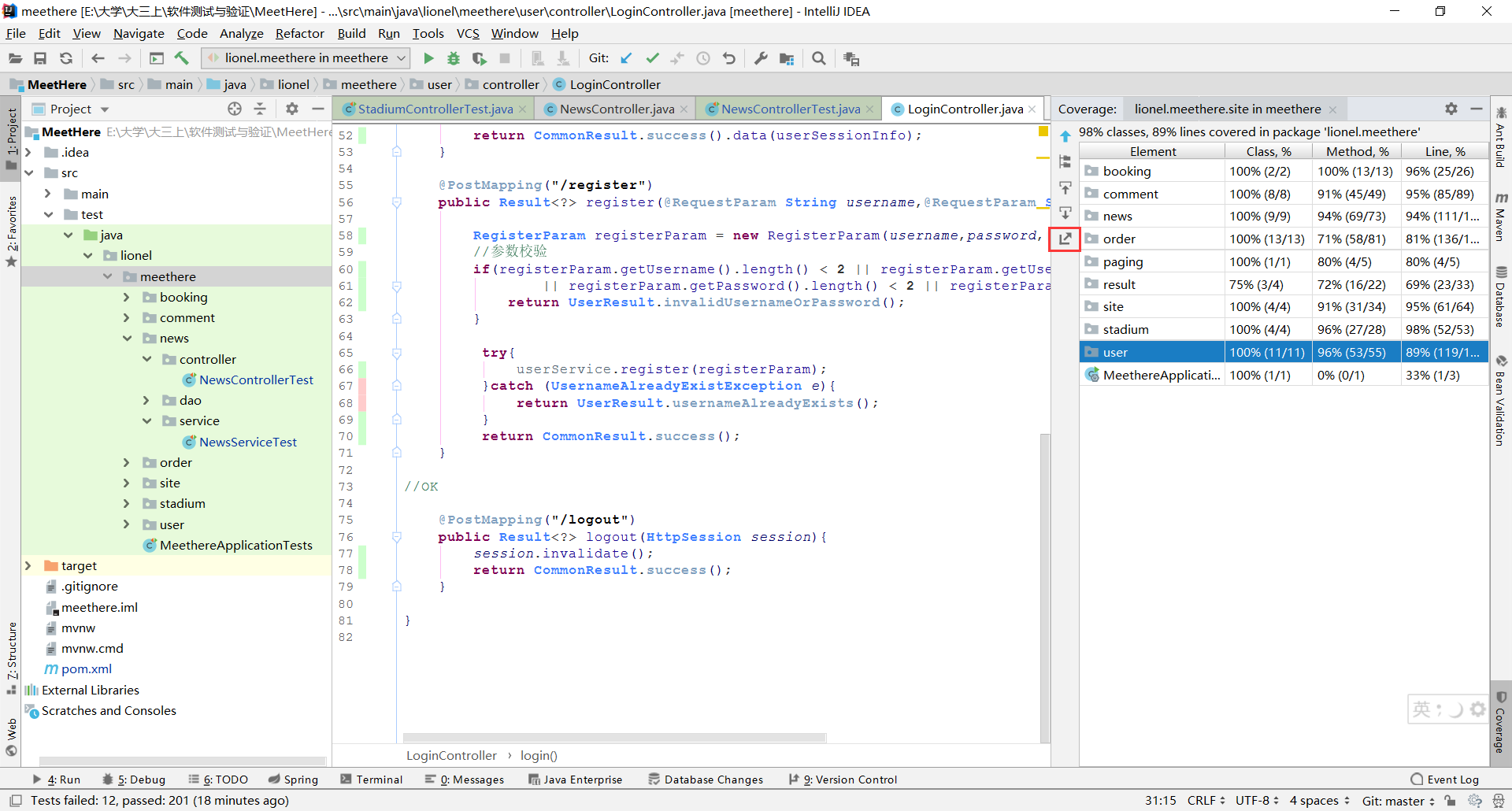
idea更强大的地方可以看到哪一行是否覆盖，如图所示绿色部分表示已经覆盖的地方，红色部分表示单元测试还没有覆盖的行。



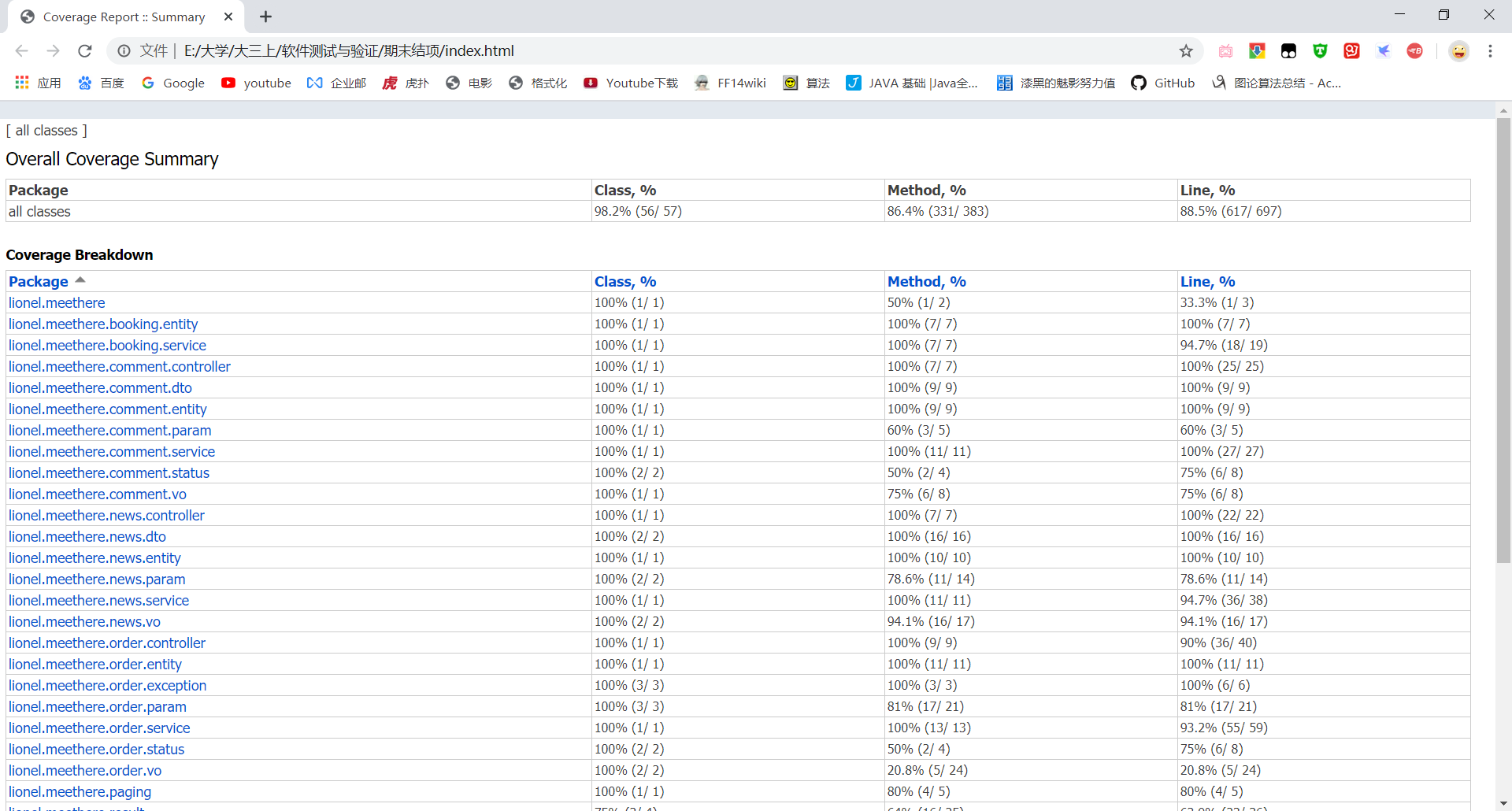
**LoginController类的代码覆盖度情况**

如图所示，我们在LoginController这个类中并没有对分支中抛出的UserNameAlreadyExistException进行测试

我们还可以将单元测试报告导出成html文件，方便交流传递



**点击红框按钮可生成html文件**



**生成的html文件**

1. **代码覆盖度的作用**

代码覆盖率的分析能在一定程度上评判代码质量，一般覆盖率高的代码出错的几率会相对低一些。但是高覆盖率只是表示执行了很多的代码，并不意味着这些代码被很好地执行了。所以，似乎覆盖率测试结果出来并不能帮我准确的评价代码质量。那么代码覆盖率对我们的开发有什么作用呢？在我们开发meethere的过程中，我们总结了代码覆盖度的以下指导意义：

1. **尽早评估代码质量**

在我们开发的过程中，定时的去看整个项目的代码覆盖度，监控覆盖报告可以帮助我们迅速找出不断增长的但是没有相应测试的代码。能够监控事情的发展，无疑是件好事。定期地查阅覆盖度报告使得设定目标(例如获得覆盖度、维护代码行的测试案例的比例等)并监控事情的发展变得更为容易。

1. **为功能测试关注点提供情报**

覆盖报告在指出没有经过足够测试的代码部分方面非常有效，因为没有被测试代码覆盖到的区域，出错的几率应该相对更高。

1. **估计修改已有代码所需的时间**

对一个开发团队而言，针对代码编写测试案例自然可以增加集体的信心。与没有相应测试案例的代码相比，经过测试的代码更容易重构、维护和增强。测试案例因为暗示了代码在测试工作中是如何工作的，所以有时还可以充当说明文档。在另一方面，没有经过相应测试的代码更难于理解和安全地修改。因此，知道代码有没有被测试，并看看实际的测试覆盖数值，可以让我们更准确地预知修改已有代码所需的时间。