**Разница между Unit testing и Integration testing.**

Именно здесь больше всего споров о названиях. «Область» интеграционных тестов также весьма противоречива, особенно по характеру доступа к приложению (тестирование в черном или белом ящике; разрешены mock-объекты или нет). На практике получается так: если тест…

* использует базу данных,
* использует сеть для вызова другого компонента/приложения,
* использует внешнюю систему (например, очередь или почтовый сервер),
* читает/записывает файлы или выполняет другие операции ввода-вывода,
* полагается не на исходный код, а на бинарник приложения,

… то это интеграционный, а не модульный тест



Подведем итог: хотя теоретически можно использовать только интеграционные тесты, на практике

* Юнит-тесты легче поддерживать.
* Юнит-тесты легко воспроизводят пограничные случаи и редкие ситуации.
* Юнит-тесты выполняются гораздо быстрее интеграционных тестов.
* Сбойные юнит-тесты легче исправить, чем интеграционные.
* Блочное ([Unit testing](http://en.wikipedia.org/wiki/Unit_testing)) — тестирование одного модуля в изоляции.
* Интеграционное ([Integration Testing](http://en.wikipedia.org/wiki/Integration_testing)) — тестирование группы взаимодействующих модулей.

Если у вас есть только интеграционные тесты, то вы впустую тратите и время разработки, и деньги компании. Нужны как модульные, так и интеграционные тесты одновременно. Они не взаимоисключающие.