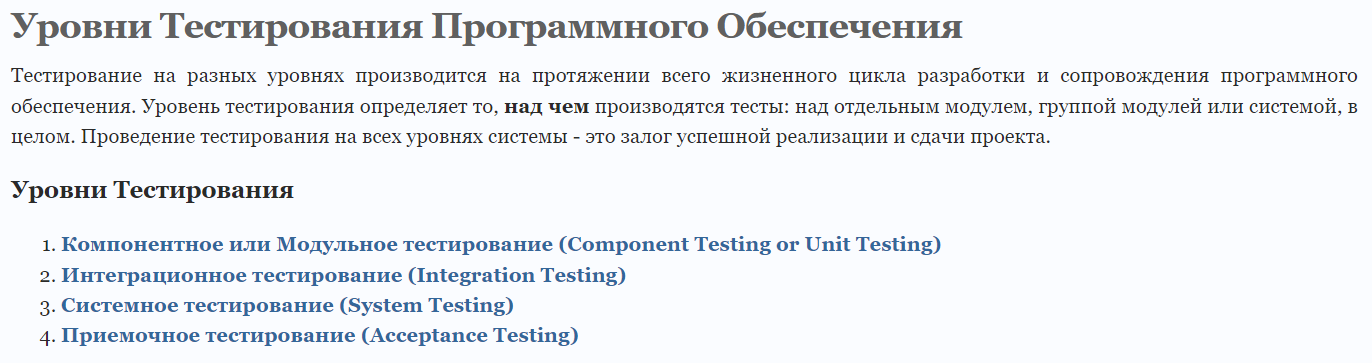
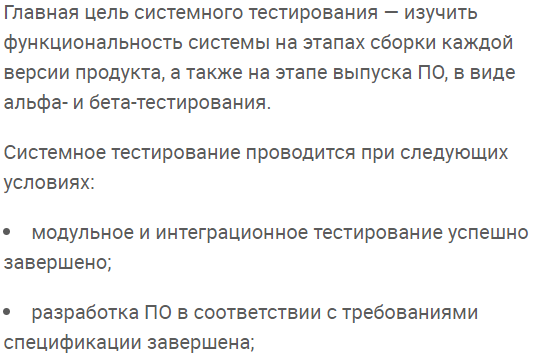
1. **Цели и задачи интеграционного тестирования. Расположение фазы интеграционного тестирования в последовательности тестов; предшествующие и последующие виды тестирования ПО.**

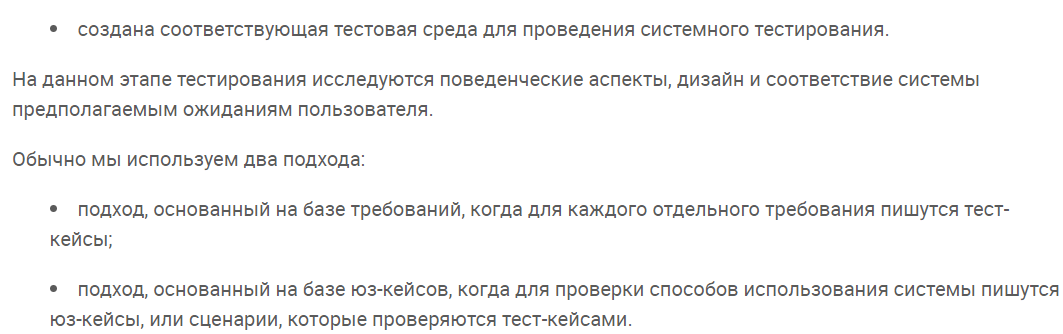
**Интеграцио́нное тести́рование** (англ. *Integration testing*, иногда называется англ. *Integration and Testing*, аббревиатура англ. *I&T*) — одна из фаз тестирования программного обеспечения, при которой отдельные программные модули объединяются и тестируются в группе. Обычно интеграционное тестирование проводится после модульного тестирования и предшествует системному тестированию.

Интеграционное тестирование в качестве входных данных использует модули, над которыми было проведено модульное тестирование, группирует их в более крупные множества, выполняет тесты. Целью тестирования является выявление багов при взаимодействии между этими программными модулями и в первую очередь направлен на проверку обмена данными между этими самими модулями.

Поскольку, как правило, модули разрабатываются разными специалистами, их понимание и логика программирования могут отличаться. Тут интеграционное тестирование становится необходимым для проверки взаимодействия модулей между собой.



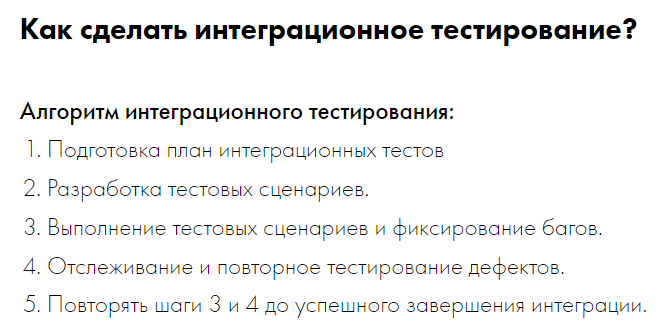




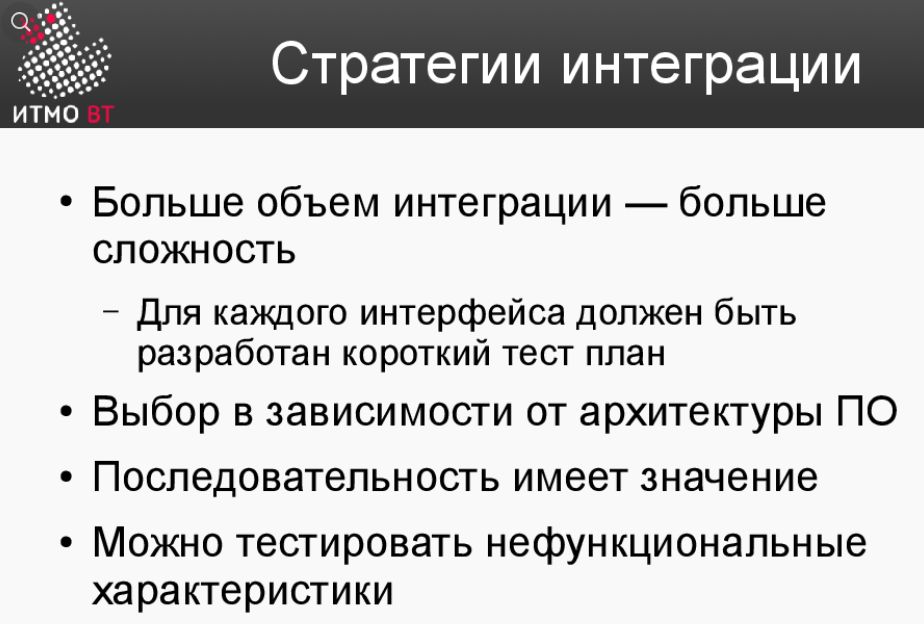
Функциональное/нефункциональные тестирования.

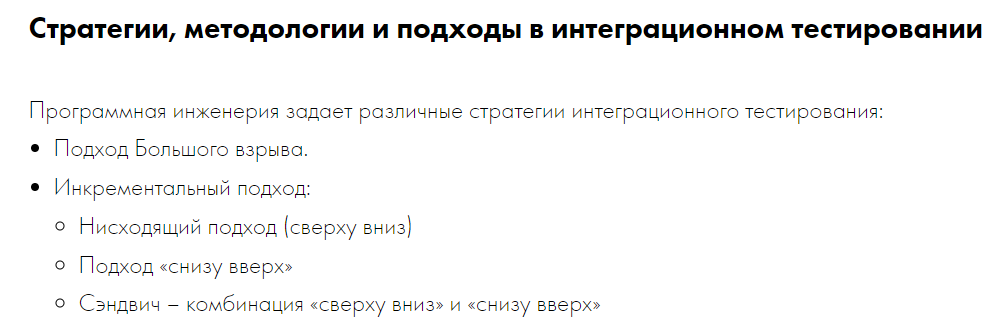


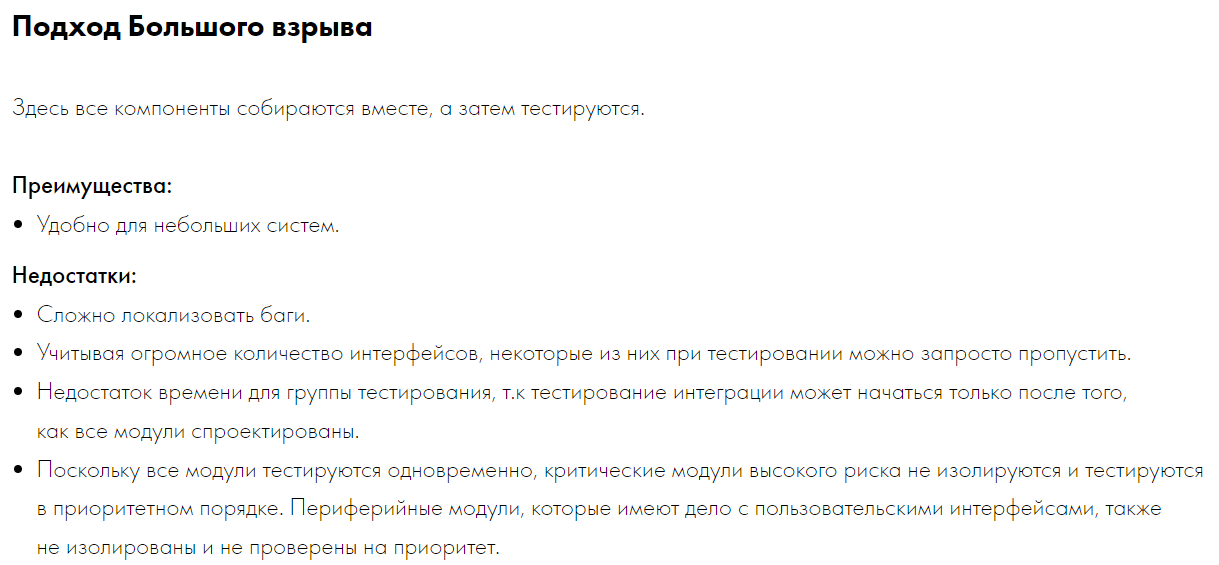
1. **Алгоритм интеграционного тестирования.**

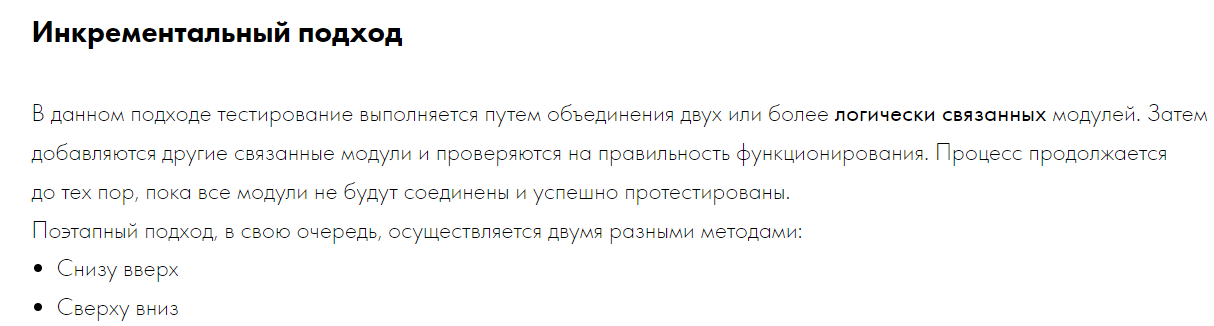


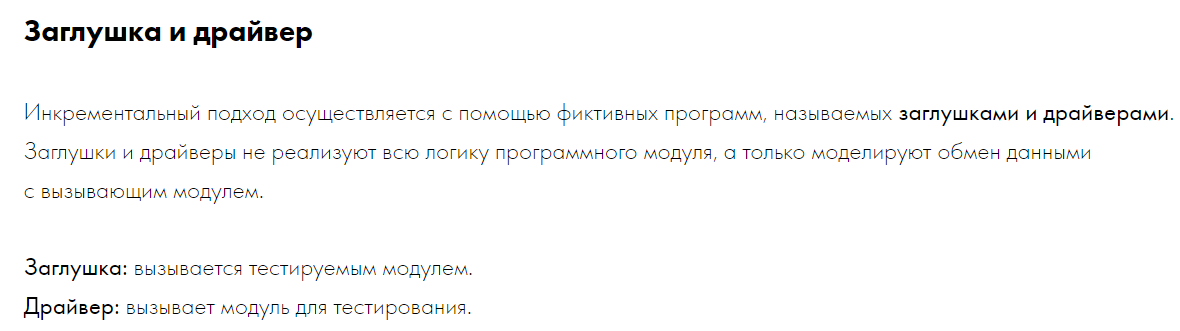
1. **Концепции и подходы, используемые при реализации интеграционного тестирования.**

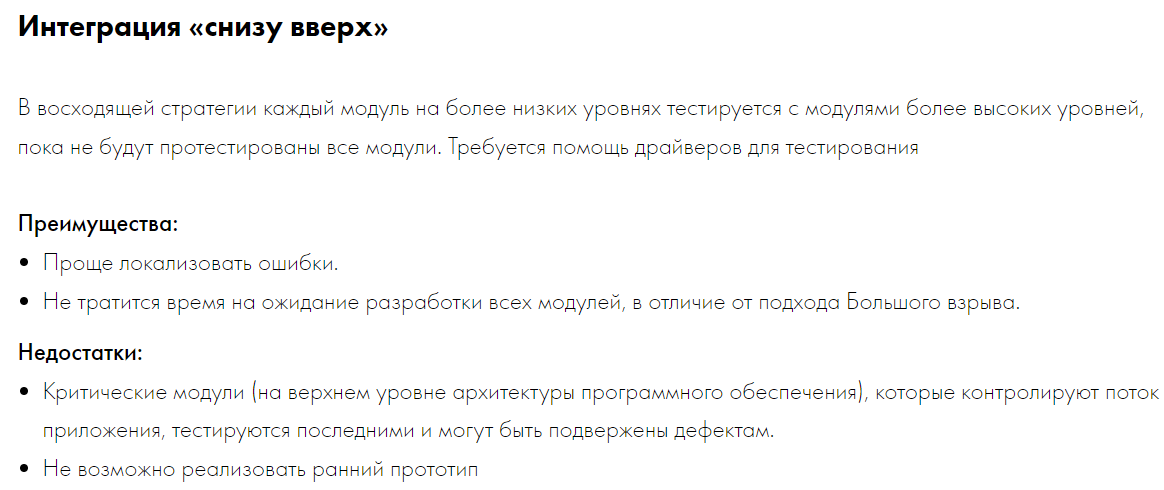


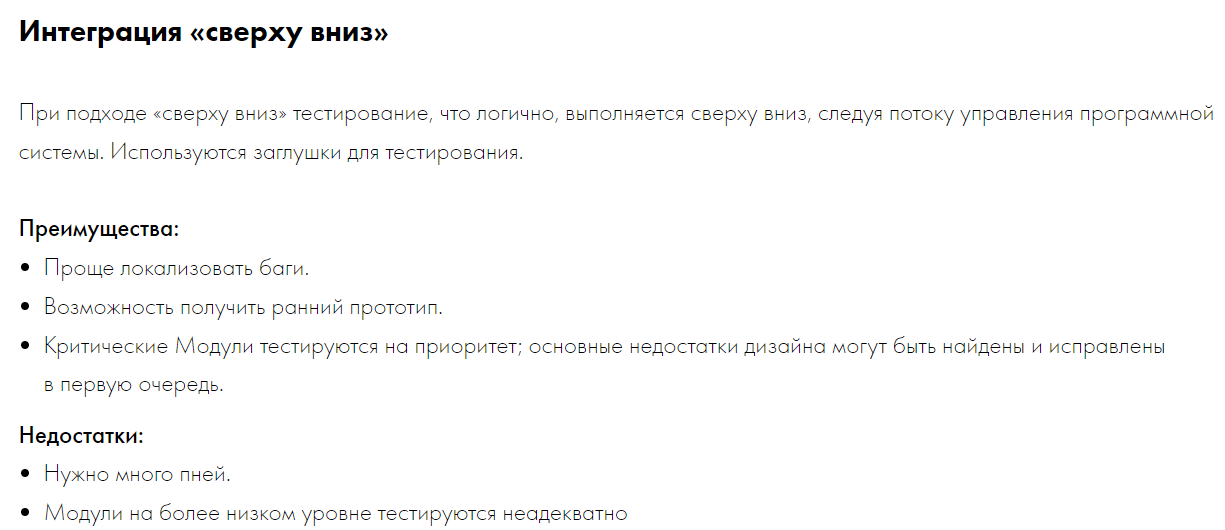


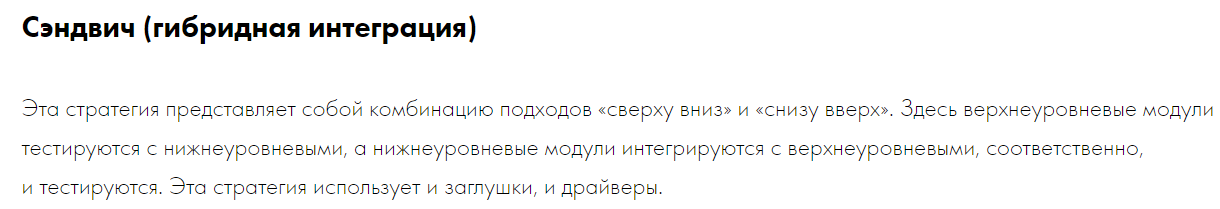


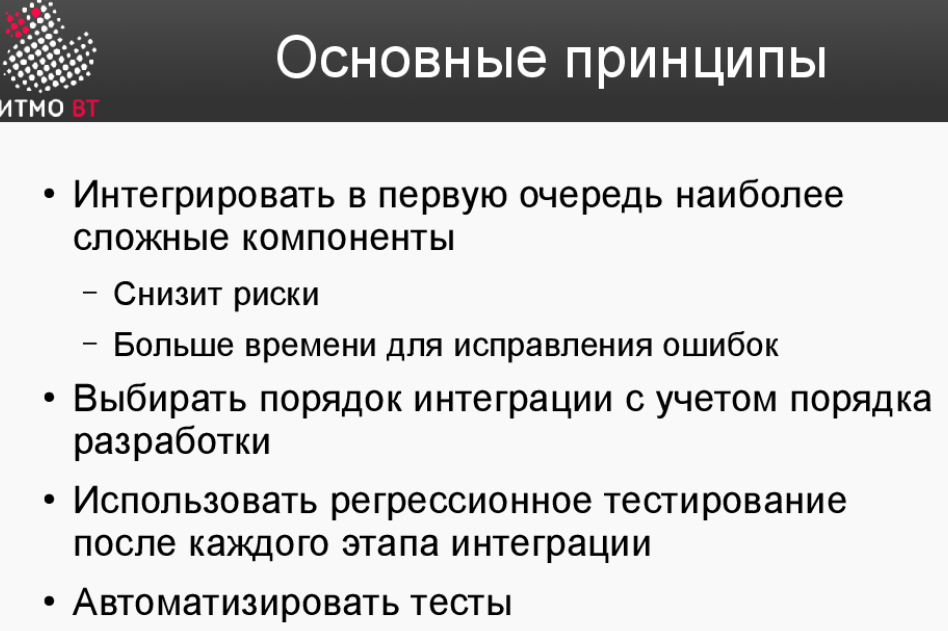




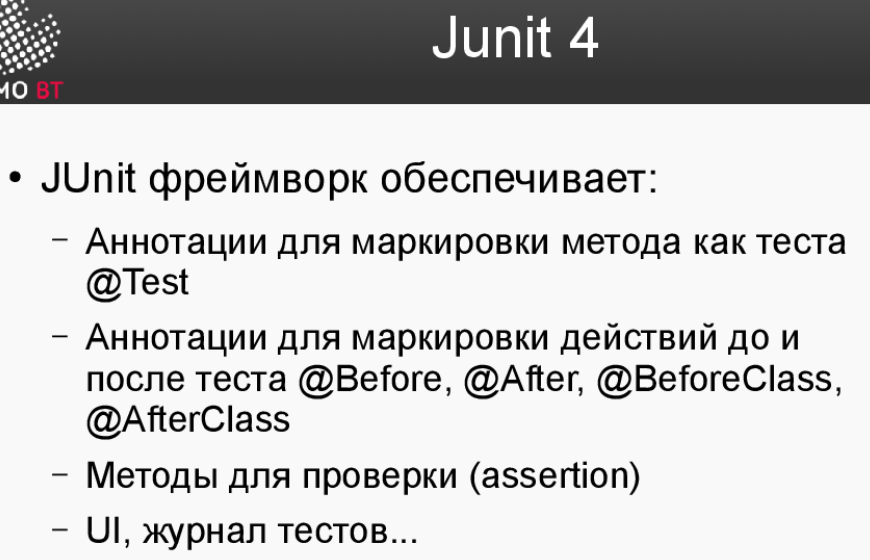


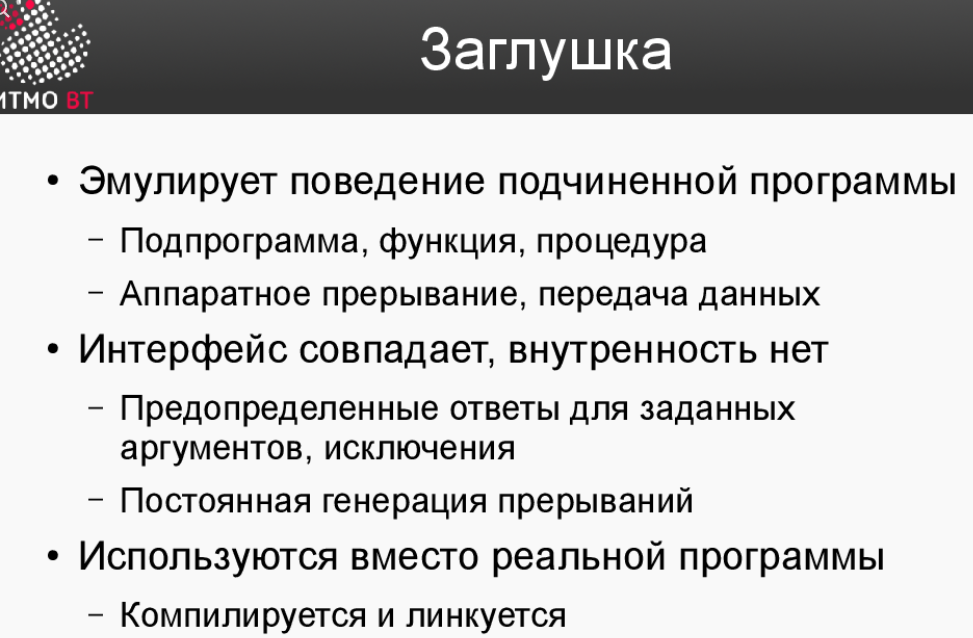


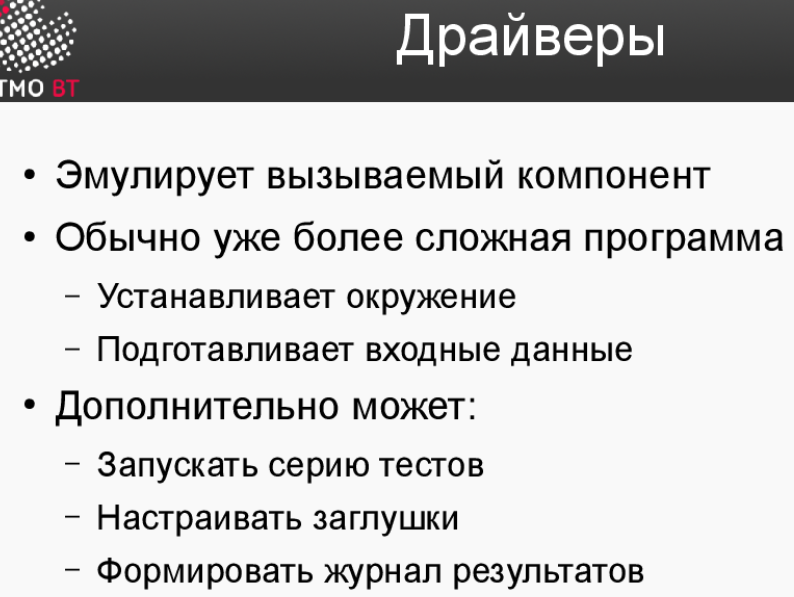


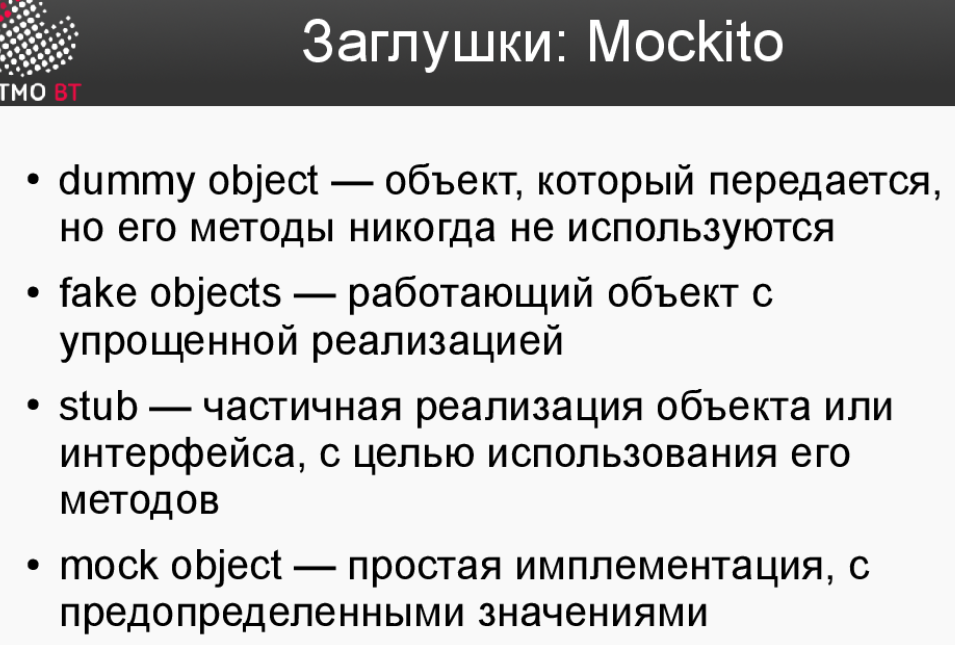


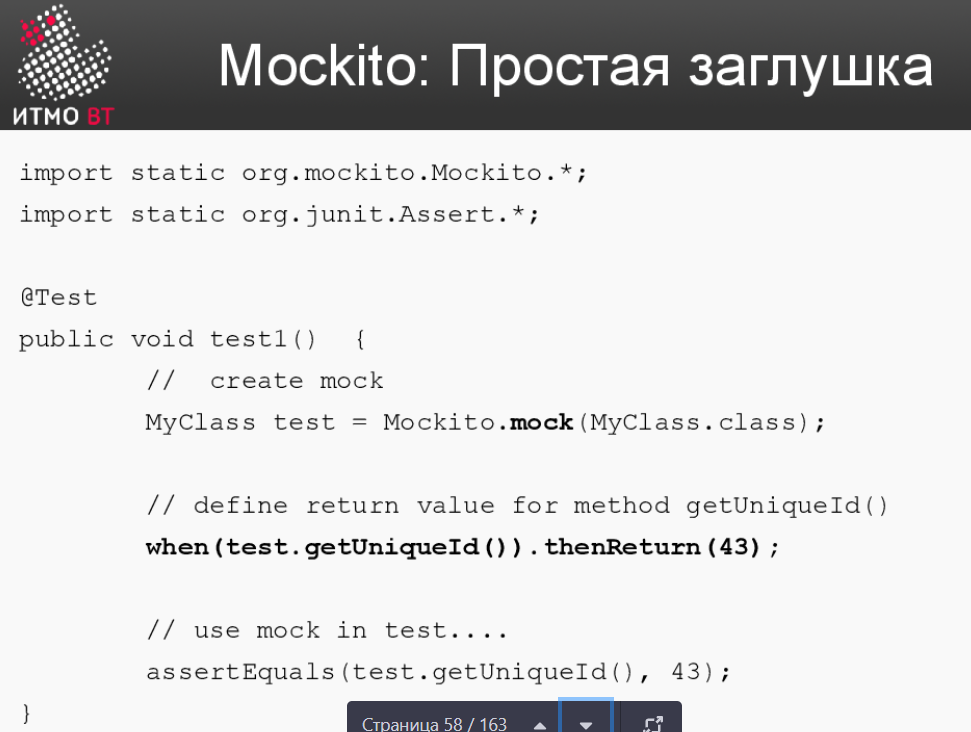
1. **Программные продукты, используемые для реализации интеграционного тестирования. Использование JUnit для интеграционных тестов.**

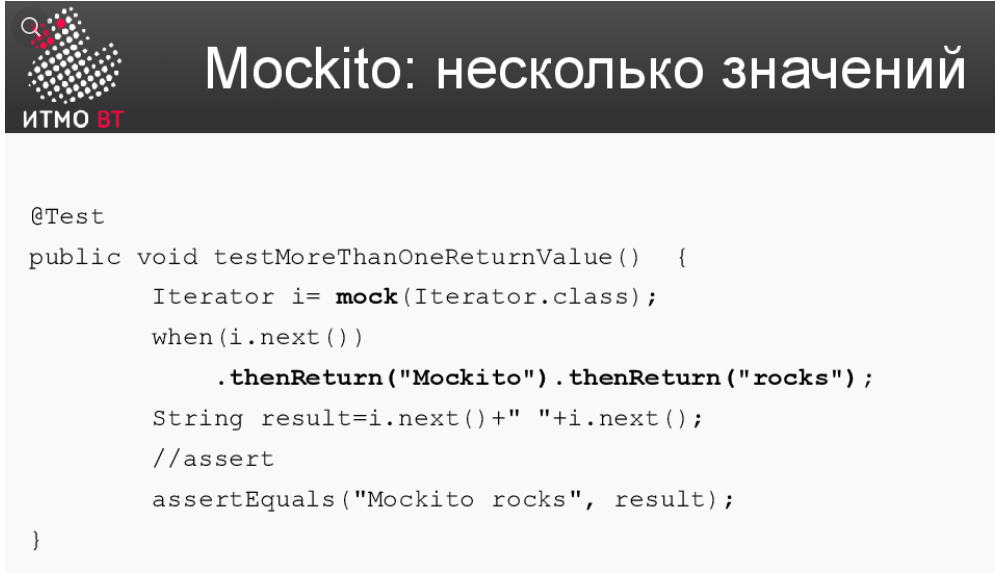


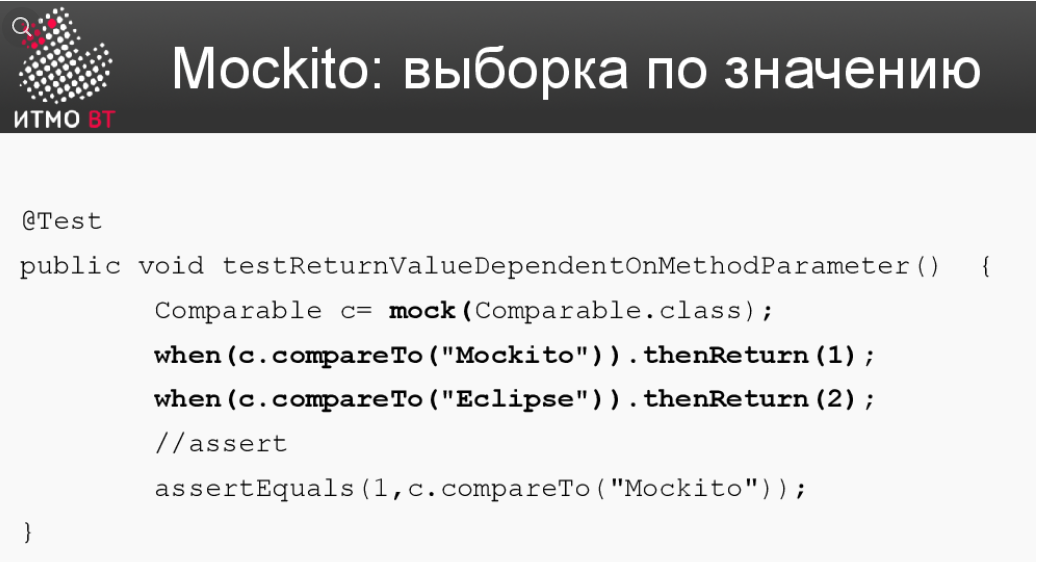


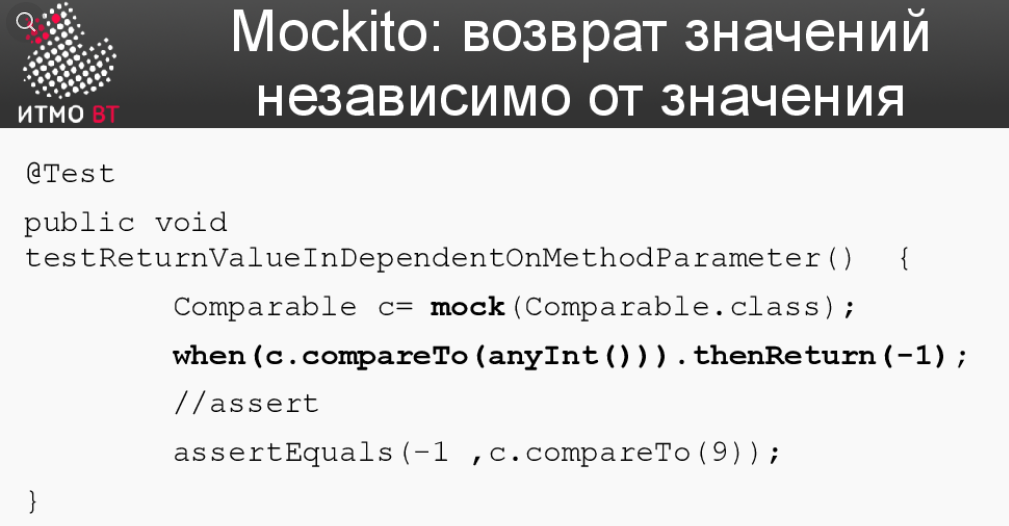


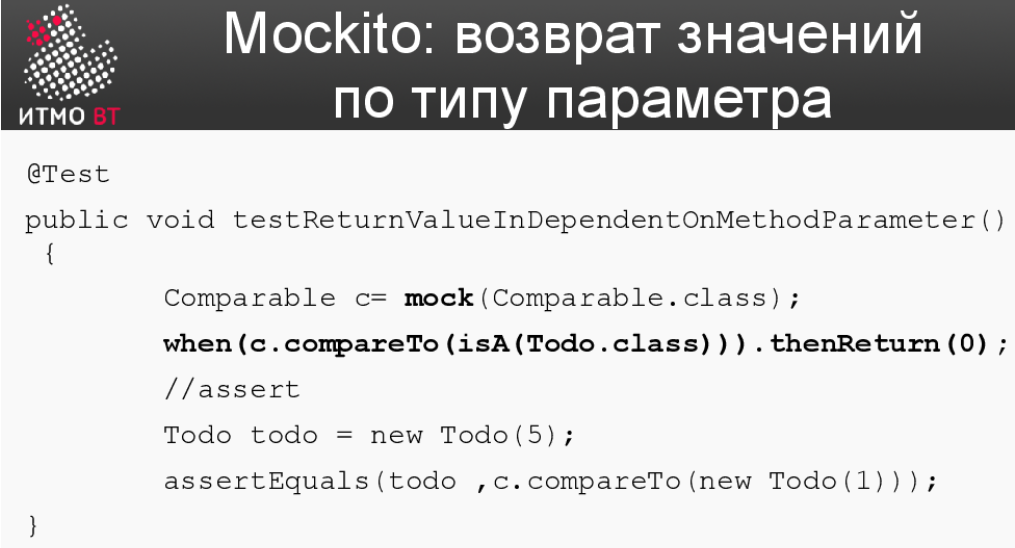






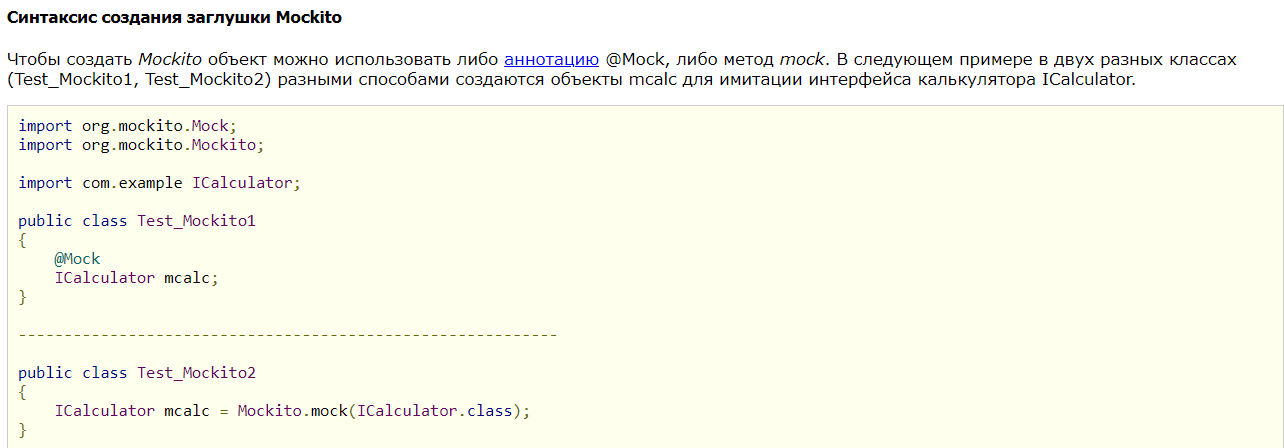


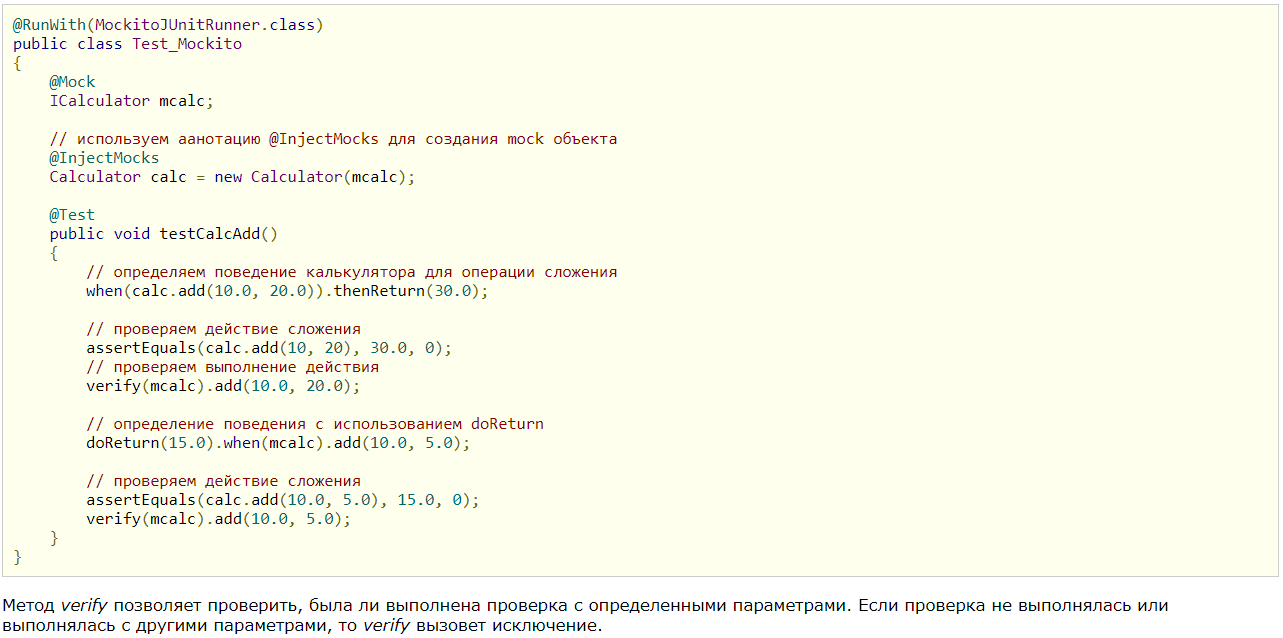


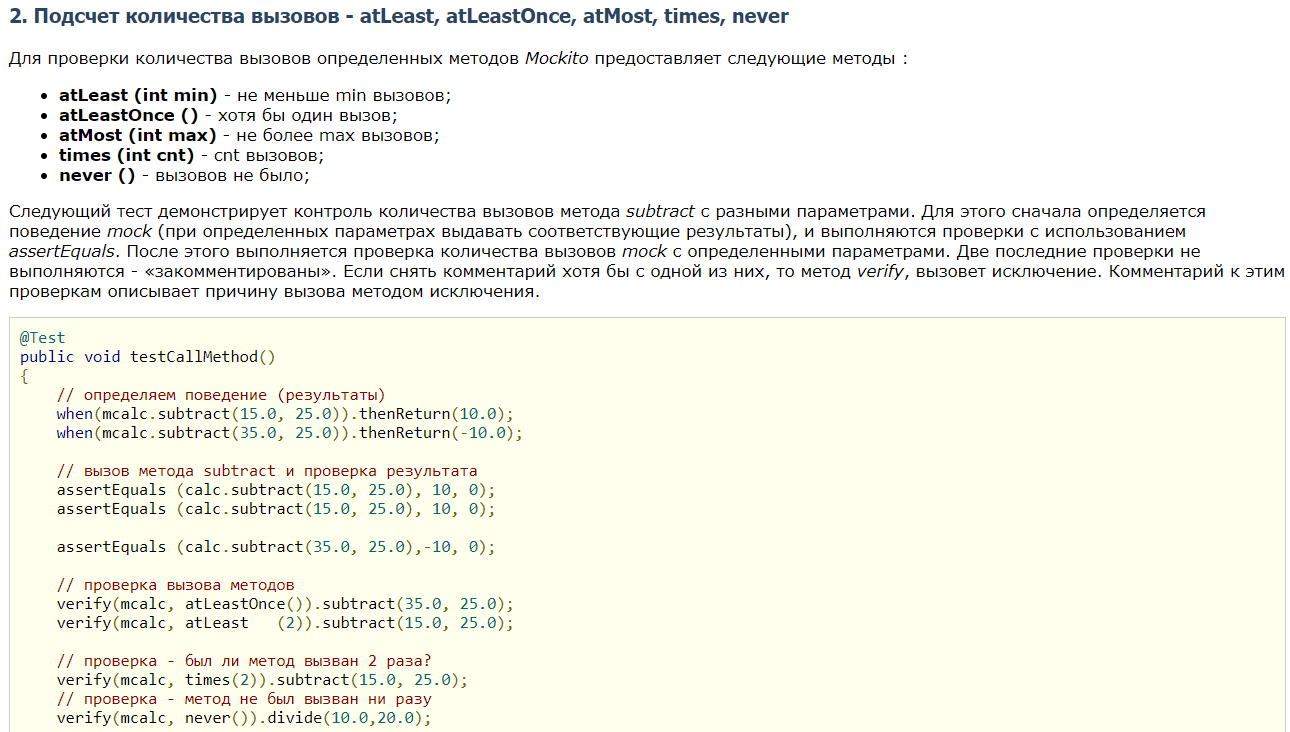


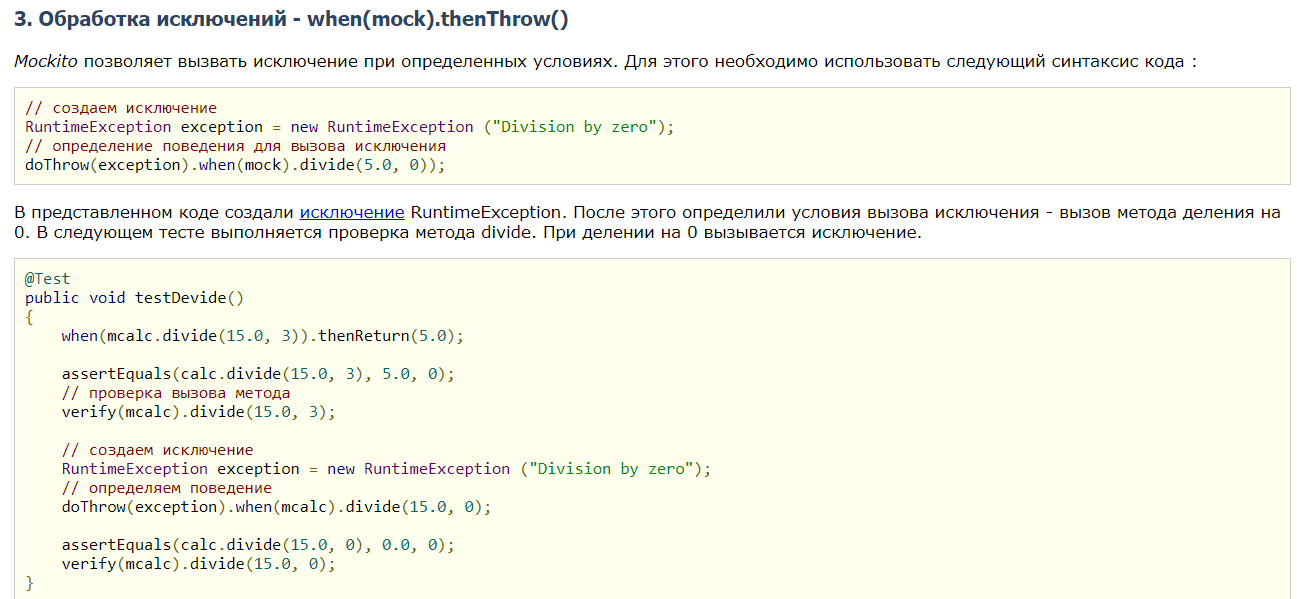
Наибольшее распространение получили следующие возможности *Mockito* :

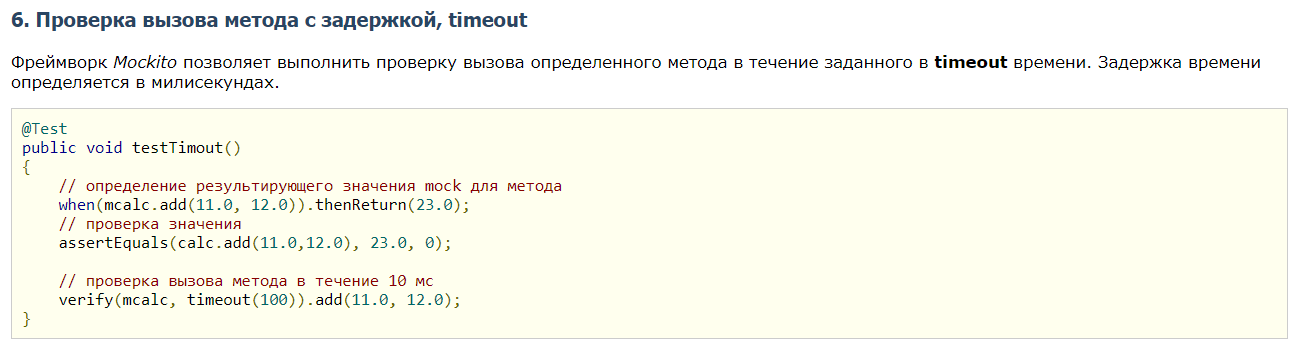
* создание заглушек для классов и интерфейсов;
* проверка вызыва метода и значений передаваемых методу параметров;
* использование концепции «частичной заглушки», при которой заглушка создается на класс с определением поведения, требуемое для некоторых методов класса;
* подключение к реальному классу «шпиона» *spy* для контроля вызова методов.

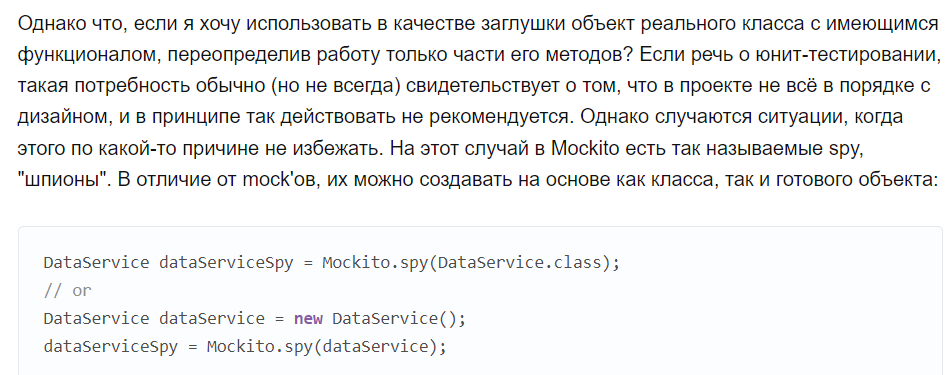








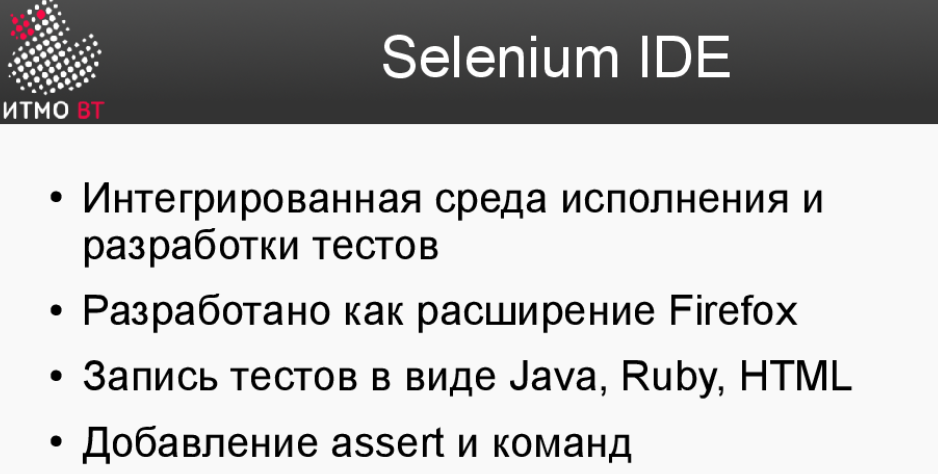


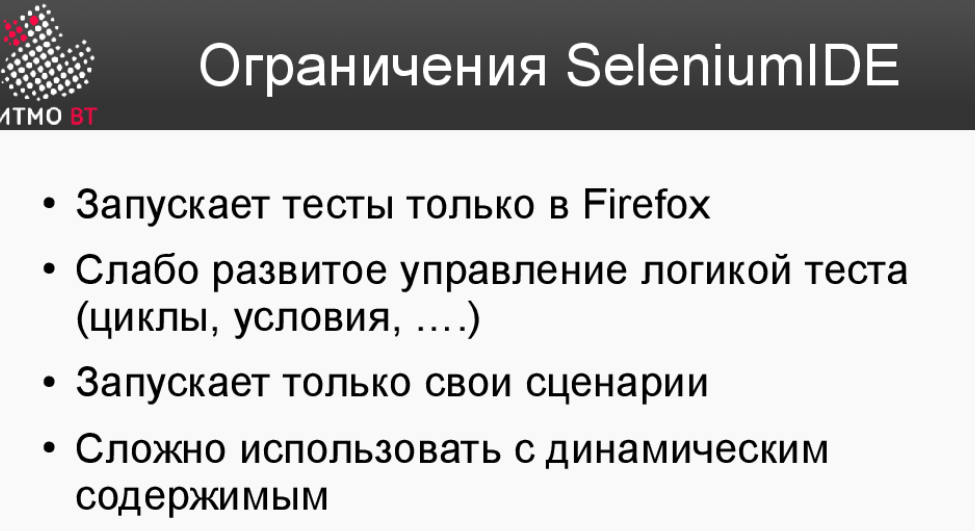


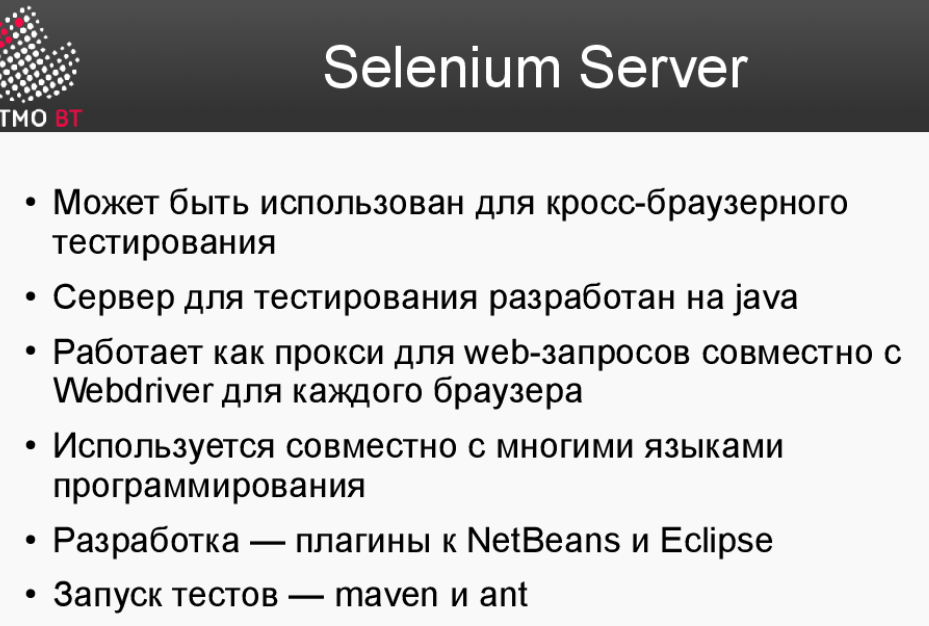
Аннотация @Spy

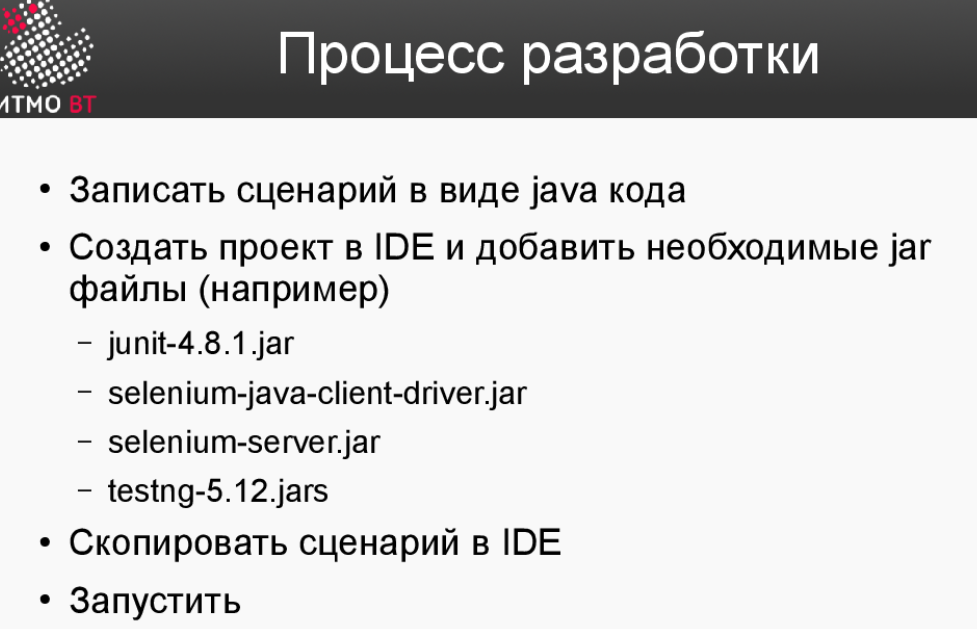
1. **Автоматизация интеграционных тестов. ПО, используемое для автоматизации интеграционного тестирования.**











Можно гонять тесты в разных браузерах + в виртуальной среде выполнения.

<https://robotdreams.cc/blog/247-kak-pisat-avtotesty-s-selenium>