# Тестирование приложений на .NET

# Когда не надо писать тесты?

- Вы делаете простой сайт-визитку из n статических html-страниц и с одной формой отправки письма
- В вашем сайте / приложении, нет никакой логики, только представления, основанные на статических файлах
- Вы делаете проект для выставки
- Вы всегда пишете код без ошибок, обладаете идеальной памятью и даром предвидения. Ваш код настолько крут, что изменяет себя сам, вслед за требованиями клиента. Иногда код объясняет клиенту, что его требования — говно

# Узнали Чака в последнем варианте?

Любой долгосрочный проект без надлежащего покрытия тестами обречен рано или поздно быть переписанным с нуля

# Какими должны быть тесты?

- Быть достоверными
- Не зависеть от окружения, на котором они выполняются
- Легко поддерживаться
- Легко читаться и быть простыми для понимания
- Соблюдать единую конвенцию именования
- Запускаться регулярно в автоматическом режиме

# По «степени изолированности кода»

- Модульное (Unit testing)
- Интеграционное (Integration testing)
- Приемочное (Acceptance testing)
- Системное (System testing)

# Модульное

- Цель модульного тестирования — изолировать отдельные части программы и показать, что по отдельности эти части работоспособны
- Тесты для каждой нетривиальной функции или метода

# Интеграционное

- Проверяет, что модули или классы правильно взаимодействуют друг с другом, при этом используется уже принцип черного ящика

# Приемочное

- Позволяет определить работает ли приложение так, как ожидает клиент

# Системное

Тестирование ПО в целом

# TDD

Test-driven development или разработка через тестирование — это техника программирования, в соответствии с которой написание кода происходит по следующему алгоритму:

1. Пишем тест
2. Тест не работает
3. Пишем код
4. Тест работает
5. Рефакторим
6. Проверяем, что тест еще работает