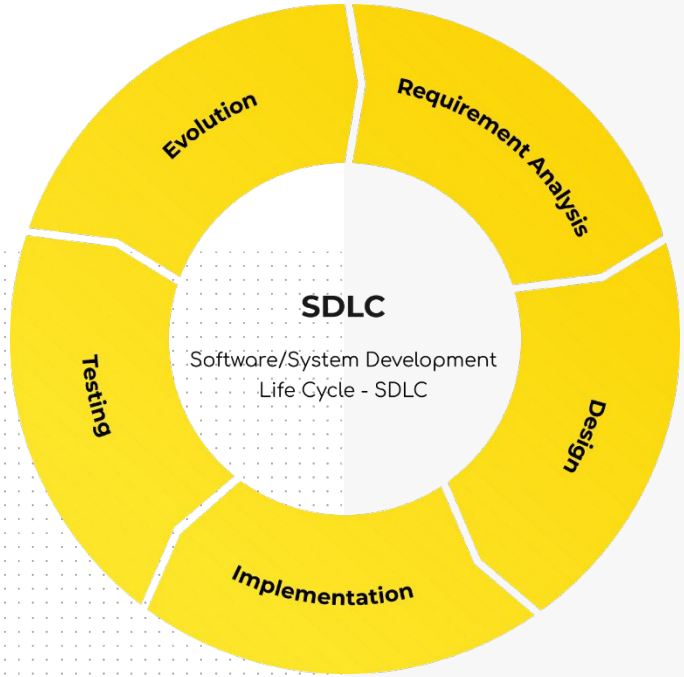
Teach Me Skills

Основы тестирования программного обеспечения

Жизненный цикл ПО (SDLC)



Начало

- Собираем требования заказчика
- Формируем бизнес-требования
- Пишем функциональные требования

Что делают QA?

Как только появляются первые документы – тестировщики начинают их анализировать и готовиться к тестированию

Уточнение

- Разработчики
  - пишут отдельные модули и юнит - тесты
- Тестировщики
  - Проводят модульное и интеграционное тестирование
- Обе группы специалистов
  - активно уточняют, дорабатывают требования и дизайн

Разработка

- Программисты
  - пишут главные функции продукта
- Тестировщики
  - тестируют функциональность на всех уровнях тестирования (пиковая активность в конце фазы)

Передача заказчику

- Команда поддержки
  - развертывает систему у заказчика
- Сторонние тестировщики/заказчик
  - проводят приемочное тестирование/UAT
- Тестировщики / программисты
  - активность спадает

Области ответственности QA

1

Планирование тестов

- разработка методологии и плана тестирования
- участие в принятии стандарта качества
- разработка того как будут выглядеть тесты/тестирование

2

Создание и выполнение тестов

- создание ручных и автоматизированных тестов
- выполнение тестов
- управление билдами (оценка состояния проекта)

3

Отчеты по тестам

- сообщить проектной группе о качестве продукта
- отслеживать состояние дефектов

Какие бывают тесты?

Представим ручку. Давайте подумаем, как её можно протестировать?

Упс. А что мы тестируем?

Тесты на основании требований

- Пользователь может поменять стержень
  - Извлекается и вставляется ли в ручку стержень
- Ручку можно зацепить за карман
  - Присутствует ли держатель, позволяющий цеплять ручку за край кармана?
- Ручка с кнопкой и возможностью спрятать стержень в корпус
  - Переключается ли ручка из рабочего в нерабочее положение?

Функциональные тесты

- Вставить в ручку стержень
- Взять ручку в руки
- Переключить в рабочее положение
- Написать несколько слов
- Переключить в нерабочее положение
- Раскрутить корпус
- Извлечь стержень

Сценарные тесты

Как ручку может использовать:

- Секретарь
- Преподаватель
- Студент
- Прораб
- Сантехник

Негативные тесты

- Что произойдет, если препятствовать выходу стержня в рабочее положение?
- Какое усилие и где надо приложить к ручке, чтобы её сломать?
- Если стержень застрял, легко ли его извлечь?
- Что произойдет, если писать по стеклу, асфальту

Тесты интерфейса

- Измерения: высота, ширина, длина, вес
- Цвет
- Читаемость логотипа фирмы-производителя
- Материал

Тесты удобства использования

- Как быстро пользователь понимает, как пользоваться ручкой?
- Как быстро пользователь привыкает к этой ручке?
- Легко ли понять, какие стержни подходят к ручке?
- Легко ли заменить стержень?
- Может ли ручкой пользоваться левша?
- Не смазываются ли чернила, если пишет левша?

Стресс-тесты

- При какой температуре расплавится пластиковая часть ручки?
- При какой температуре потечет стержень?
- При какой температуре ручка перестает писать?
- Какое воздействие нужно применить к ручке, чтобы сломать её?
- Пишет ли ручка под водой? А по мокрой бумаге?
- Если ручку уронить в песок – что произойдёт?
- А если уронить со стола?
- А если из окна офиса?

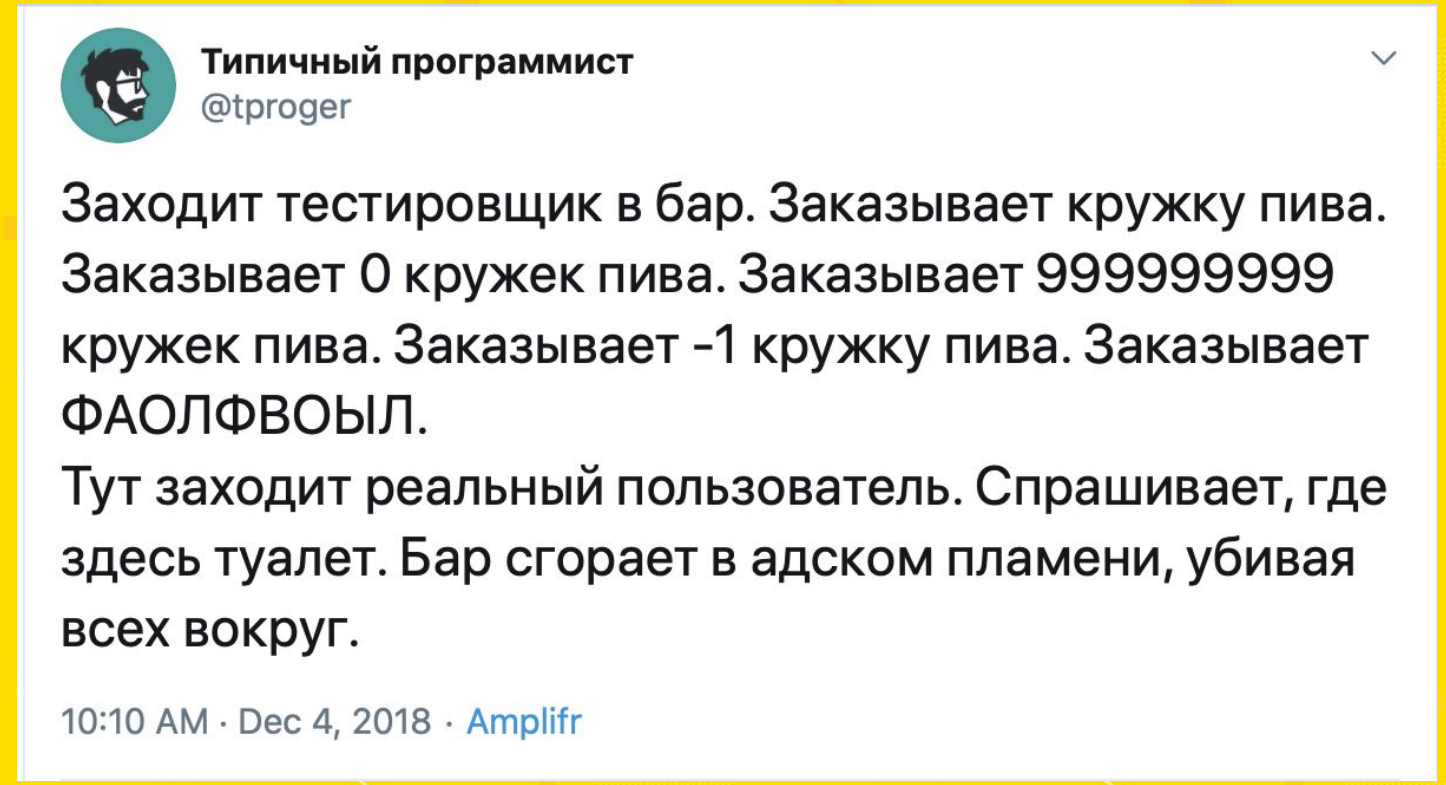
Тесты производительности

- Сколько текста можно написать ручкой в единицу времени?
- Как быстро ручку можно привести в рабочее положение?
- Как много раз ручку можно переключить из нерабочего в рабочее положение, прежде чем её начнёт заедать?

Конфигурационные тесты

- Какие стержни подходят к нашей ручке?
- На каких поверхностях она может писать

Но не забывайте о главном...



Типичный программист
@tproger

Заходит тестировщик в бар. Заказывает кружку пива. Заказывает 0 кружек пива. Заказывает 999999999 кружек пива. Заказывает -1 кружку пива. Заказывает ФАОЛФВОЫЛ.
Тут заходит реальный пользователь. Спрашивает, где здесь туалет. Бар сгорает в адском пламени, убивая всех вокруг.

10:10 AM · Dec 4, 2018 · Amplifr

Чек-лист

Чек-лист (checklist) - список проверок без описания шагов; упрощенная форма тест-кейса

- у него нету четкой структуры, вернее их очень много
- из важных характеристик – краткость и понятность (простота), быстрота создания и понимания

Пример чек-листа

Операции с файлами	ok	
Создание файла	ok	
Открытие файла	ok	
Сохранение документа	ok	
Печать	ok	
Редактирование файлов	bugs	
Отмена	ok	
Копирование	ok	
Вырезание	ok	
Вставка	ok	
Удаление	ok	
Поиск	fail	bug #123
Поиск с заменой	fail	bug #126
Вставка даты	ok	
Форматирование	ok	
Перенос строки	ok	
Изменение шрифта	ok	

Тест-кейсы

Тест-кейс – (тестовый случай) совокупность шагов, условий и параметров созданных для проверки работоспособности функции или ее части

A set of test inputs , execution conditions , and expected results developed for a particular objective , such as to exercise a particular program path or to verify compliance with a specific requirement

(Набор тестовых входных данных, условий выполнения и ожидаемых результатов, разработанных с конкретной целью, такой как проверка некоторого пути выполнения программы или проверка соответствия некоторому требованию)

Структура тест-кейса

- Номер (number) или идентификатор (id)
- Связанное с тестом требование (related requirement)
- Модуль (Feature)
- Имя (name)
- Предусловия (Preconditions)
- Шаги (Steps)
- Ожидаемый результат (Expected result)
- Статус (Passed, Failed, Blocked)
- Приоритет (Priority: Low, Medium, High)
- Связанный с тестом баг (если есть) (related bug)
- Постусловия (Post-conditions)

Пишем тест-кейс

Название: Валидация подсчета скидки в диапазоне количества книг 20-49

Шаги:

1. Добавить в корзину 20 книг
2. Нажать ‘Proceed to checkout’
3. Проверить финальную стоимость
4. Повторить шаги 1-3 для 49 книг

Ожидаемый результат: Скидка 2% должна применяться на общую стоимость

Тест-дизайн

Классы эквивалентности

Класс эквивалентности (equivalence class) - набор тестов, со схожими входными данными, шагами воспроизведения и одним ожидаемым результатом

Класс эквивалентности – множество, все элементы которого программа обрабатывает одинаково

Признаки эквивалентности

- Если один из тестов обнаруживает ошибку, другие её тоже, скорее всего, обнаружат (и наоборот, не обнаружит один, не обнаружат все)
- Тесты используют одни и те же наборы входных данных
- Для выполнения мы совершаем одни и те же действия
- Тесты генерируют одинаковые выходные данные или приводят приложение в одно и то же состояние
- Все тесты приводят к срабатыванию одного и того же блока обработки ошибок

Граничные условия

Граничные условия (границы) - это те места, в которых один класс эквивалентности переходит в другой

Граничные условия очень важны, т.к. именно в этом месте чаще всего и будут ошибки