

Программа

Введение

1. Примеры наиболее дорогих ошибок
2. Причины возникновения:
 - Космические лучи :)
 - Ошибки в ПО

Виды тестирования. Версионирование

1. Различные классификации
 - По цели
 - По свойствам
 - По исполнителю
 - По уровню
 - По интерфейсу
2. Тестирование требований?
3. Пирамида тестирования:
 - Приоритеты различных видов тестирования
 - Соотношение видов тестов
4. Принципы семантического версионирования
 - Стандарт версионирования
 - Пример на разделяемом протоколе взаимодействия
 - Пример на библиотечных решениях
5. Политика версионирования при тестировании:
 - Альфа-версии
 - Бета-версии

Планирование тестирования (Test case / Bugs)

1. Структура и назначение Test-plan:
 - Кто
 - Что
 - Как
 - Когда
 - Критерии
2. Структура и назначение Test-case:
 - Предусловия
 - Шаги
 - Фокусирование на функциональности
3. Заведение ошибок:
 - Workflow
 - Основные поля и принципы их заполнения
 - Поиск дубликатов по стекам

4. Вовлечённость тестировщика в работу команды

Test design

1. Black Box
 - Классы эквивалентности
 - Граничные значения
 - Доменный анализ
 - Диаграмма переходов состояний
 - Попарное тестирование
 - Тестирование вариантов использования
2. White Box
 - Потоки управления
 - Потоки данных
3. Experience based
 - Checklists
 - Exploratory
 - Error Guessing

Test design. Exploratory testing

- Function
- Claims
- Domain
- User
- Stress
- Risk
- Flow
- Automatic
- Scenario

Тестирование API

1. Особенности тестирования протоколов
 - RPC
 - Messaging
2. Уровни протоколов
3. Форматы сериализации:
 - JSON
 - SOAP
 - Protobuf
4. Архитектура генерации кода для сервера и клиента и mock для них

Тестирование API. Wireshark

1. Архитектура ядра ОС и снифферов
2. Захват простого TCP-трафика в Wireshark
3. Захват широковещательного UDP-трафика
 - Фильтрация по порту
4. Анализ пропускной способности сети
5. Просмотр Flow graph
6. Пример Decode as для преобразования RDP → RTP
7. Захват и анализ HTTP трафика
 - Выделение TCP-коммуникаций запроса и ответа

Тестирование API. Postman

Теория

1. Напоминание принципов протокола HTTP (GET/POST/...)
2. JSON-schema/OpenAPI/Swagger - основы и примеры описания REST API
3. Аналогия с XSD/WSDL

Практика

1. Postman:
 - Импорт описания API
 - Применение окружений
 - Создание запросов
 - Использование переменных и их переопределение
 - Тесты на JS для проверки:
 1. Кода возврата
 2. Полей ответа
 3. Соответствия схеме
 - Назначение и применение mock-серверов

Fuzzing-тестирование

Теория

1. Пример ГОСТ 56939-2024
2. Виды верификации:
 - Статическая (SonarQube, PVS)
 - Динамическая (asan, ubsan, fuzzing)
3. Sanitizers:
 - asan
 - ubsan
4. Архитектура Fuzzing:

- Начальная выборка
 - Контроль трасс исполнения
 - Генетические алгоритмы
5. Критерии остановки тестирования
 6. Аналогия с Reinforcement Learning

Практика

1. AFL fuzzer:
 - Контейнерный запуск
 - Создание тестовых данных для затравки
 - Пример на дереве условных операторов
 - Запуск afl-fuzz и разъяснение полей, выводимых в runtime
 - Разбор результата поски падения приложения

Тестирование API. SoapUI

Теория

1. XSD-схемы - основы и примеры описания типов
2. WSDL-сервис - основы и пример описания методов

Практика

1. SoapUI:
 - Создание проекта на основе WSDL
 - Посылка запросов и получение ответов
 - Создание TestSuit
 - Проверки на основе XPath

Нагрузочное тестирование

Теория

1. Фокусы тестирования тестирования производительности:
 - Нагрузка
 - Стабильность
 - Отказоустойчивость
 - Масштабируемость
 - Стресс-тестирование
2. Профили нагрузки:
 - SLA
 - Пределы производительности
3. Параметры:
 - Время обработки

- Чатота запросов
- Размер данных
- 4. Откуда брать профили нагрузки:
 - БД
 - Журналы
 - Прогноз
- 5. Инструменты:
 - Web-консоль
 - JMeter
 - Gatling
 - K6

Практика

1. JMeter:
 - Поддерживаемые протоколы
 - Ручное создание HTTP-запросов
 - Запись сценариев через Proxy
 - Thread group и его параметры
 - Вынесение общих параметров
 - Просмотр результатов в графическом и табличном видах

Тестирование интерфейса пользователя. Web/Desktop

Теория

1. Классификация по технологиям:
 - Desktop
 - Web
2. Классификация инструментов по завязке на технологию разработки
3. Примеры технологий разработки интерфейса и соответствие инструментов тестирования со знанием идентификаторов элементов интерфейса:
 - Qt: Squish
 - JS: Selenium
 - Java: Assert4
4. Применение компьютерного зрения: Sikuli
 - Архитектура
 - OpenCV
 - Tesseract
 - Jython
5. Применение чистого машинного обучения: Testolang
 - Архитектура
 - QEMU/KVM
 - Нейронные сети
6. Применение AI-агентов:
 - Архитектура
 - MCP и Playwright
 - Проблема контекста DOM

7. Архитектура Selenium:

- WebDriver
- API на Python, Java, ...
- IDE как расширение браузеров

Практика

1. Selenium:

- Создание виртуального окружения на Python
- Запуск WebDriver
- Поиск элементов на странице (css, id, атрибуты)
- Ввод текстовых данных
- Автоматизированная генерация сценария в IDE

Тестирование на проникновение

Дополнительные темы в зависимости от квалификации курса:

1. docker (3 часа)
2. git (3 часа)

From:

<https://se.moevm.pro/> - МОЭВМ Вики [se.moevm.pro]

Permanent link:

<https://se.moevm.pro/doku.php/courses:testing:lectures>

Last update:

