



Тестовая документация. Виды. Тест-кейсы, чек-листы.
Техники тест-дизайна

Курс: «Мануальное тестирование ПО»

Процесс тестирования.

1

Выбрать действие

Выбрать ожидаемый результат этого действия

2

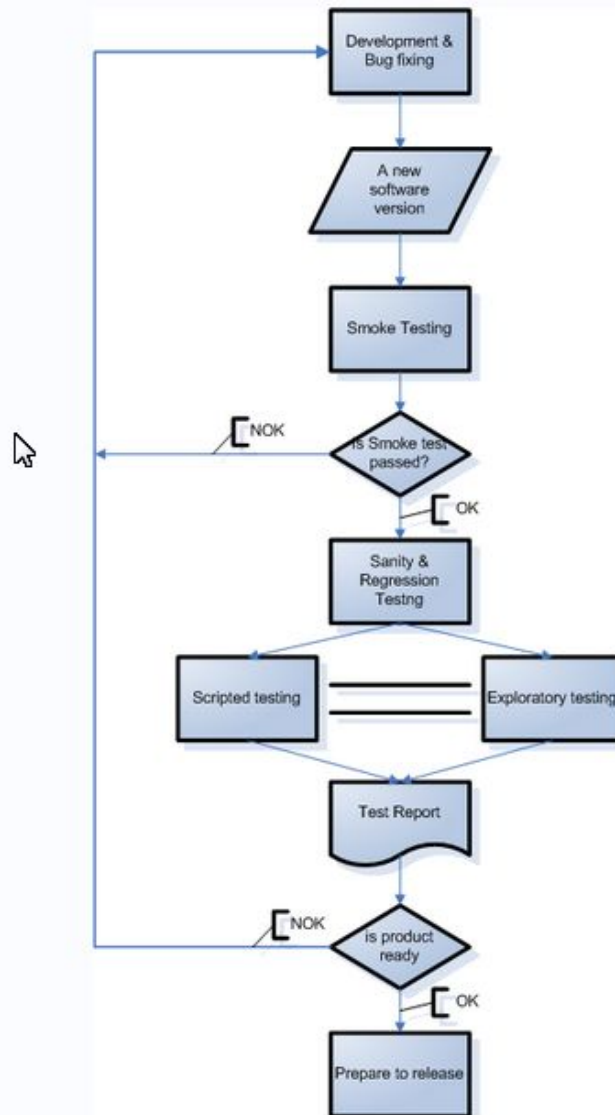
3

Узнать фактический результат

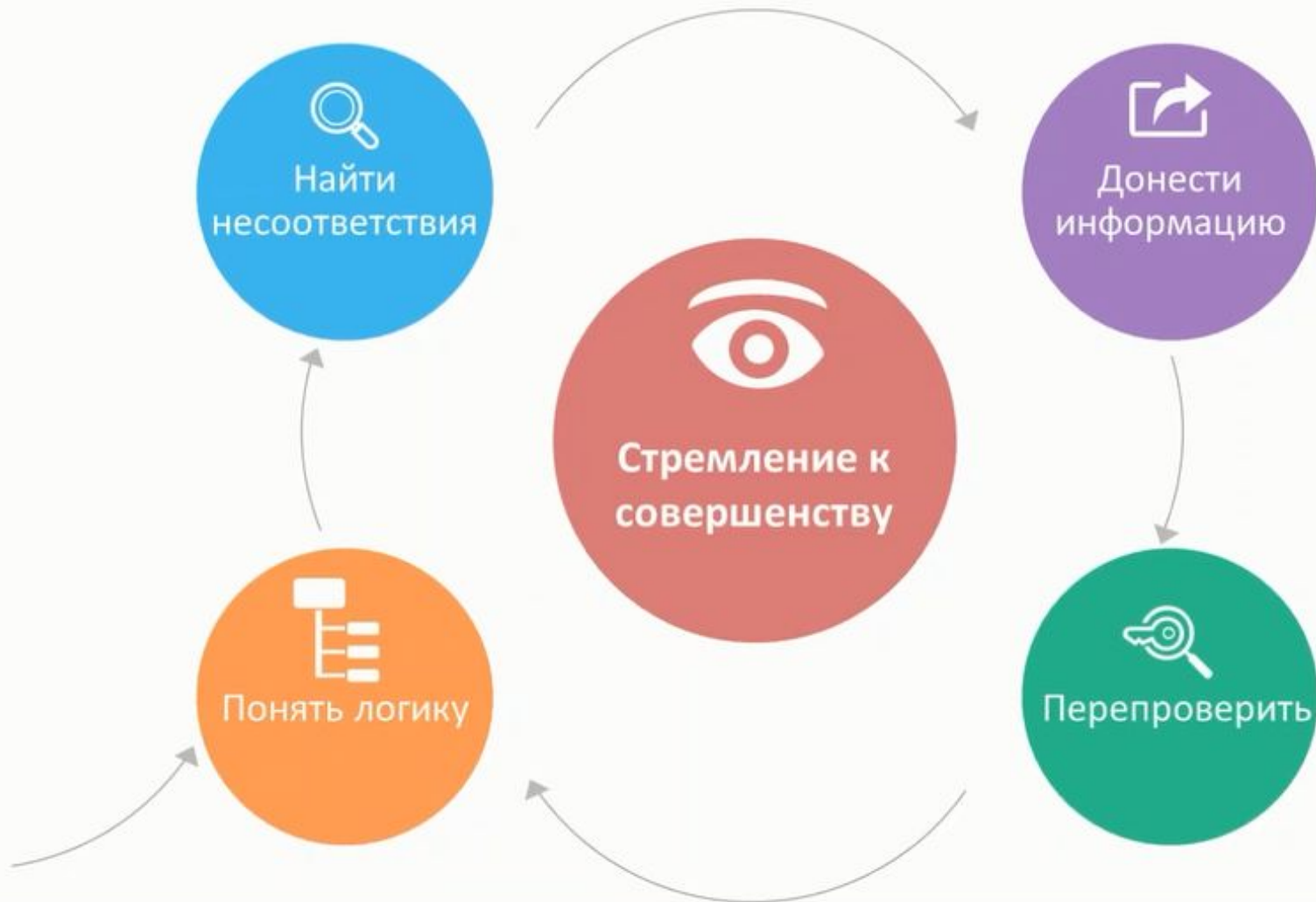
Сравнить эти результаты

4

Процесс тестирования.



Вечный круг тестирования.



Тестовые артефакты (тестовая документация).

- **Спецификация программного обеспечения** (Software Specification), Требования (Модуль 4)
- **План тестирования** (Test Plan) (Модуль 3)
- **Чек лист** (Check-list)
- **Тестовый случай** (Test Case)
- **Тестовый набор** (Test Suite)
- **Баг Репорты** (Bug Reports) (Модуль 6)
- **Отчеты о тестировании** (Test Results Reports)

Чек лист

- **Чек лист (Check-list)** - список шагов или перечень функциональностей, который позволяет тестировщику убедиться в корректной работе приложения.

Пример:

Проверка работы регистрационной формы	Результат
Зарегистрировать нового пользователя с логином содержащим максимальное количество символов	Passed
Зарегистрировать нового пользователя с именем %%% и пробелами	Failed
Зарегистрировать пользователя "admin", и пользователя "admin" (где а – из русской раскладки)?	Passed

Check-list

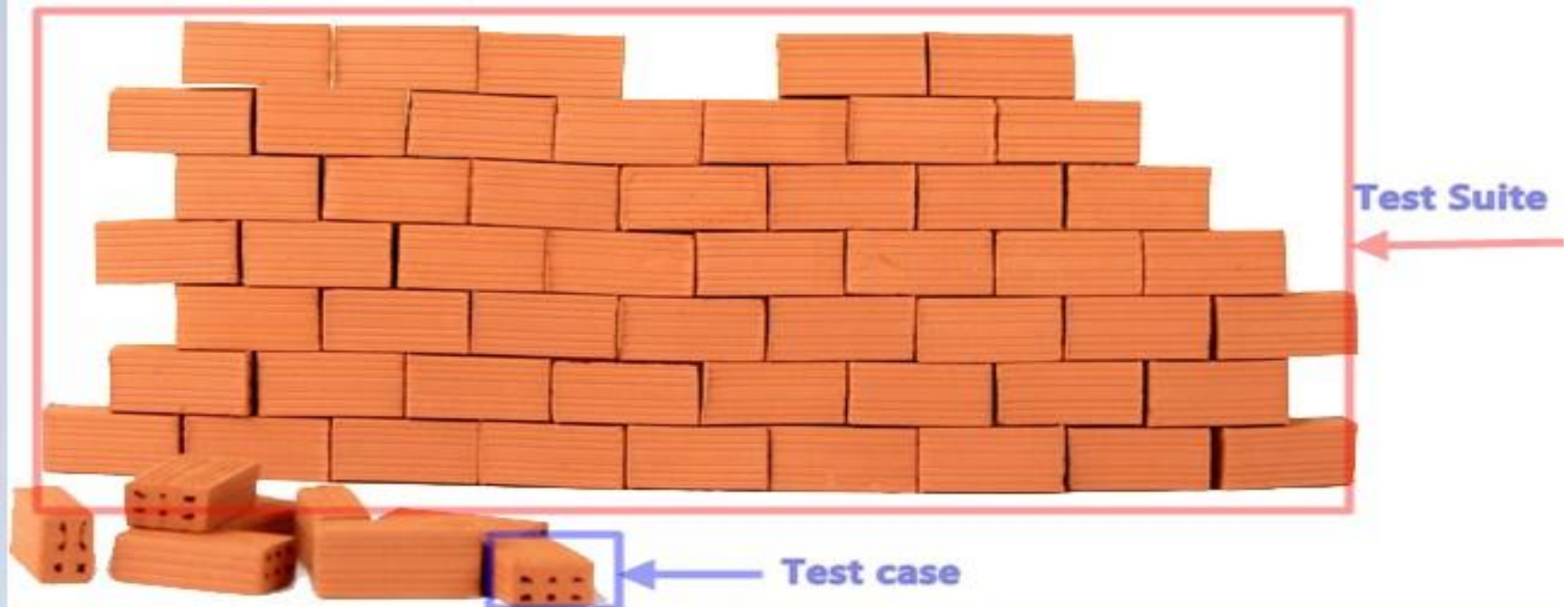
ver.	Result
Catalogs	
Создание пустого каталога	ok
Создание каталога с вещами	x
Создание вещи\каталога\списка без символов в имени	?
Выбор вещей в каталоге (Android only)	?
Переименование каталога	?
Добавление вещей в существующий каталог	?
Выбор вещей в каталоге (название в неск.строк) (проверка checked) (Android only)	?
Редактирование вещей в каталоге	?
Создать каталог с уже существующим именем	?
Создание списка посредством выбора вещей\создать список без выбора вещей (Android only)	?
Создание списка посредством выбора вещей(список с уже существующим именем)(Android only)	?
Добавление вещи с уже существующим именем	?
Чувствительность к регистру(Вещи, каталоги)	?
Создать каталог\вещь с использованием спец. символов и пробелов(!""№;%:?*()_+/\,.)	?
Создать каталог\вещь с использованием цифровых символов(1234568790)	?

Определения

- **Тест дизайн (Test Design)** – это этап процесса тестирования ПО, на котором проектируются и создаются тестовые случаи (Test Case), в соответствии с определёнными ранее критериями качества и целями тестирования.
- **Тестовый набор (Test Suite)** - это набор тестов реализующих бизнес-задачу, выполняемую тестируемой системой. Тестовый набор включает в себя, кроме тестовых сценариев, ещё и тестовые данные и правила их генерации.
- **Тестовый случай (Test Case)** - это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Test Suite

- На примере **Test Suite** можно рассмотреть так:
- Test Suite - это кирпичная стена,
- Test Case – это один кирпич из стены.
- В Test Suite попадают тест кейсы объединённые по какой либо роли, функциональности.



Тест кейс.

- **Тестовый случай (Test Case)** - это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Action	Expected Result	Test Result (passed/failed/blocked)
Нажать на кнопку "Войти".	Происходит переход на страницу "Authentication".	Passed

Тест кейс должен быть унифицированным – в рамках одного тест кейса использовать одни и те же термины, обозначения.

Тест кейс должен быть однозначным и понятным - значение каждой фазы и слова, должно пониматься в единственно возможном смысле. Не используйте слова «плохо», «хорошо», «очевидно».

Тест кейс не должен быть слишком простым или слишком сложным, не используйте длинных, запутанных сложноподчинённых предложений (лучше разделить один шаг на несколько).

Тест кейс должен проверять одну функциональность и содержать до 10-ти шагов.

Тест кейсов не должно быть слишком много, т.к. их потом трудно будет поддерживать.

Тест кейсов не должно быть слишком много, т.к. их потом трудно будет поддерживать.

Тест кейсы должны быть независимыми друг от друга

Не дублируйте одинаковые шаги в разных тест кейсах.
Сокращайте, или выносите в предусловия.

Виды тестовых случаев.

- Позитивный тест кейс (пользователь вводит корректные данные)
- Негативный тест кейс (пользователь вводит корректные данные)

Структура тест-кейсов

- Каждый тест кейс имеет 3 основные составляющие:
- - **PreConditions** (Предусловия) – список действий, которые приводят систему к состоянию пригодному для проведения основной проверки. Либо список условий, выполнение которых говорит о том, что система находится в пригодном для проведения основного теста состоянии.
- - **Test Case Description**(Описание тестового случая) – список действий, переводящих систему из одного состояния в другое, для получения результата, на основании которого можно сделать вывод о удовлетворении реализации, поставленным требованиям .
- - **PostConditions** (Постусловия) –список действий, которые возвращают систему в первоначальное состояние.
- Примечание: Post Conditions не является обязательной частью. Эта часть актуальна при автоматизированном тестировании, когда за один прогон можно наполнить базу данных множеством некорректных документов. Поэтому желательно возвращать систему в первоначальное состояние.

Поля, которые содержит тест-кейс

- **ID** (уникальный номер теста)
- **Epic** (module) (модуль системы, к которому относится данный тест)
- **Summary** (краткое описание, ИДЕЯ тестового случая)
- **Description** (подробное описание того, что будет тестироваться)
- **Steps** (подробное описание шагов)
- **Data** (используемые в процессе тестирования, данные)
- **Expected result** (ожидаемый результат)
- **Actual result** (Фактический результат)
- **Requirement #** (ссылка на требование)
- **Bug #** (ссылка на баг)
- **Criticality** (важность тестового случая)
- **Comment** (комментарий)

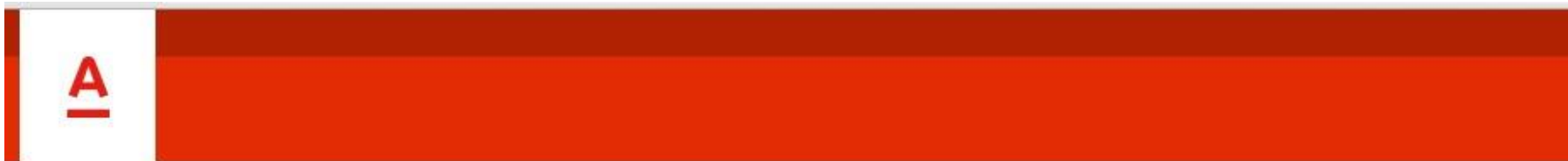
Показываем формат...

Пример тест кейса 1: Проверка отображения страницы.

Действие	Ожидаемый результат
Открыть страницу "Вход в систему"	<ul style="list-style-type: none">- Окно "Вход в систему" открыто- Название окна - Вход в систему- Логотип компании отображается в правом верхнем углу- На форме 2 поля - Имя и Пароль- Кнопка Вход доступна- Ссылка "забыл пароль" - доступна

Пример тест кейса 2: Проверка отображения страницы.

- **Действие:** Открыть страницу «Вход в систему»
Проверка: Проверьте, что отображаемая страница соответствует странице на рисунке (и прилагаем изображение)




Добро пожаловать в интернет-сервис «My Alfa-Bank»!

Пожалуйста, руководствуйтесь подсказками при регистрации и входе в систему.

Логин

Пароль

[Новый пользователь](#) [Забыли логин или пароль](#)






Безопасный и комфортный Интернет-банкинг

Советы по написанию тест кейсов:

- Разбейте функционал программы и начните составление тест кейсов для одной из её частей;
- Используйте ранее составленные чек листы для создания общей структуры тест кейсов;
- Начните с простых позитивных тестов;
- Помните о граничных значениях и классах эквивалентности;
- Добавьте негативные тесты;
- Уточните спорные моменты;
- Подумайте, какие необычные сценарии можно проверить;
- Не переходите к следующей части, пока не закончите предыдущую;
- Используйте аналогичные тесты для остальных частей.

Напишите тест кейсы для формы входа в почтовый ящик. (Demo)

 **Почта**  

<input type="text" value="имя ящика"/>	<input type="text" value="@mail.ru"/> ▼
<input type="password" value="пароль"/>	<input type="button" value="Войти"/>

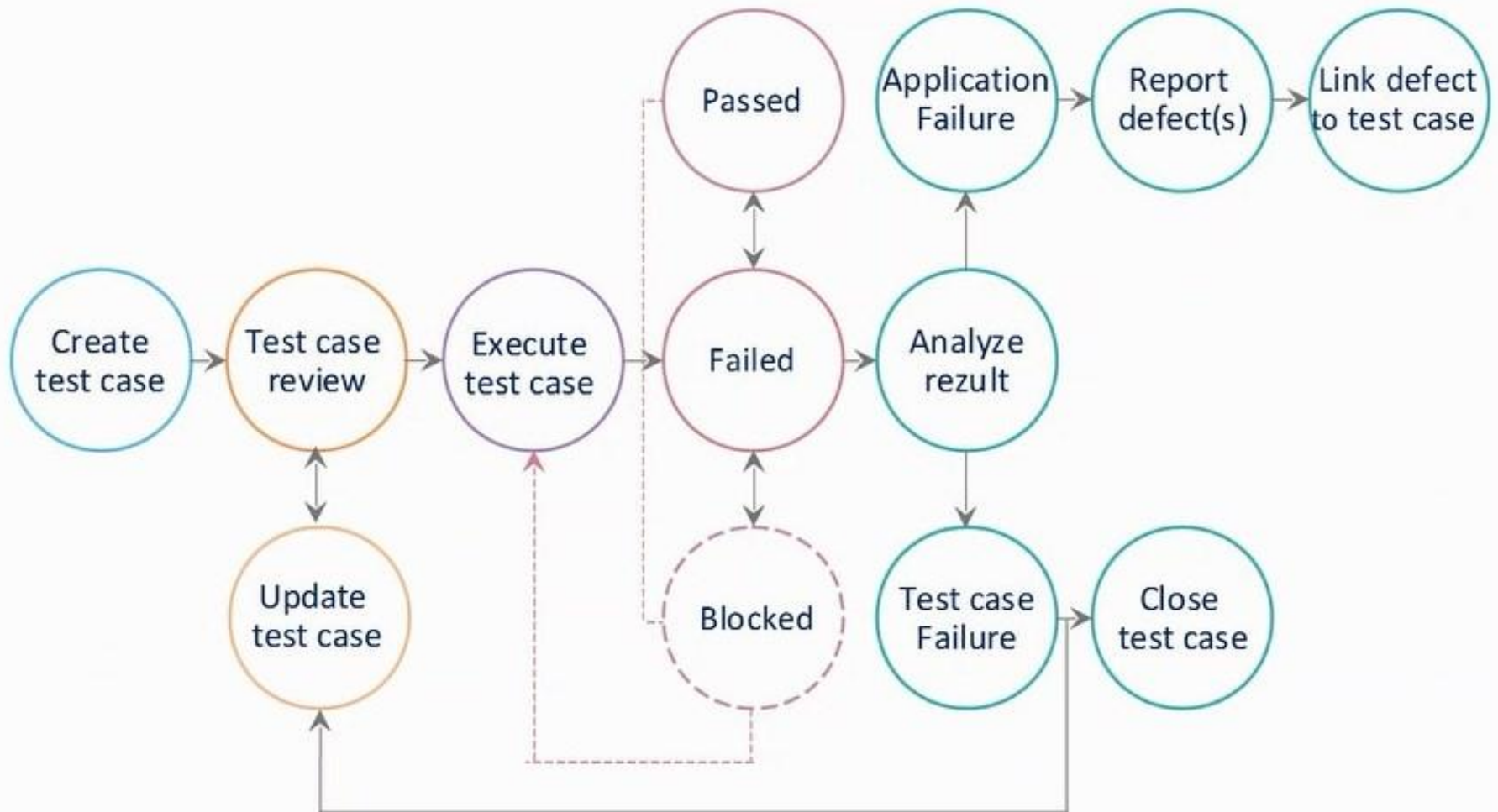
[Забыли пароль?](#) запомнить

Регистрация в почте
с просмотром картинок

Примеры тест кейсов для формы входа в почтовый ящик:

- T1: Внешний вид страницы для входа в почту.
- T2: Авторизация зарегистрированного пользователя с помощью корректных данных.
- T3: Авторизация с помощью корректного логина и некорректного пароля (и наоборот) зарегистрированного пользователя.
- T4: Авторизация с помощью некорректной пары логин-пароль
- T5: Авторизация с пустыми полями логина и пароля.

Жизненный цикл тест кейса.



Для чего нам нужны тест-кейсы/
чек-листы?



Для чего нам нужна тестовая документация

- 1. Передача знаний
- 2. Ускорение регрессионного тестирования
- 3. Помощь при автоматизации
- 4. Отчетность о выполненной работе
- 5. Визуализация покрытия требований
- 6. Оценка трудозатрат

Основные техники тест дизайна.

- Верификация, валидация
- Positive\ negative testing
- Эквивалентное разделение, классы эквивалентности
- Анализ граничных Значений (Boundary Value Analysis)
- Причина/ Следствие (Cause/Effect)
- Предугадывание ошибки (Error Guessing)
- Исчерпывающее тестирование (Exhaustive Testing)
- Попарное тестирование (Pairwise testing)
- AD НОС testing
- Дымовое (Smoke testing)

Эквивалентное разделение (Equivalence Partitioning - EP), классы эквивалентности (equivalent classes-EC)

- **Класс эквивалентности** - множество тестов со сходными параметрами, протестировав один из них, можно поставить галочку, что протестировал и все остальные (остальные параметры множества будут иметь тот же результат).
- **Эквивалентные тесты** — это тесты, которые приводят к одному и тому же результату.

Эквивалентное разделение (Equivalence Partitioning - EP), классы эквивалентности (equivalent classes-EC)

- Например, у вас есть диапазон допустимых значений от 1 до 10, Вы должны выбрать одно верное значение внутри интервала, скажем, 5, и одно неверное значение вне интервала - 15.

Анализ граничных Значений (Boundary Value Analysis)

Например, пусть мы тестируем программу для отдела кадров, в ней есть поле "Возраст соискателя".

Требования по возрасту у нас будут такие:

- 0-13 лет - не нанимать
- 14-17 лет - можно нанимать на неполный день
- 18-54 года - можно нанимать на полный день
- 55-99 лет - не нанимать

Пример взят из книги A Practitioner's guide to Software Test Design (Lee Copeland).

Анализ граничных Значений (Boundary Value Analysis)

```
if (age >= 0 && age <=13)
    hireStatus="NO";
if (age >= 14 && age <=17)
    hireStatus="PART";
if (age >= 18 && age <=54)
    hireStatus="FULL";
if (age >= 55 && age <=99)
    hireStatus="NO";
```

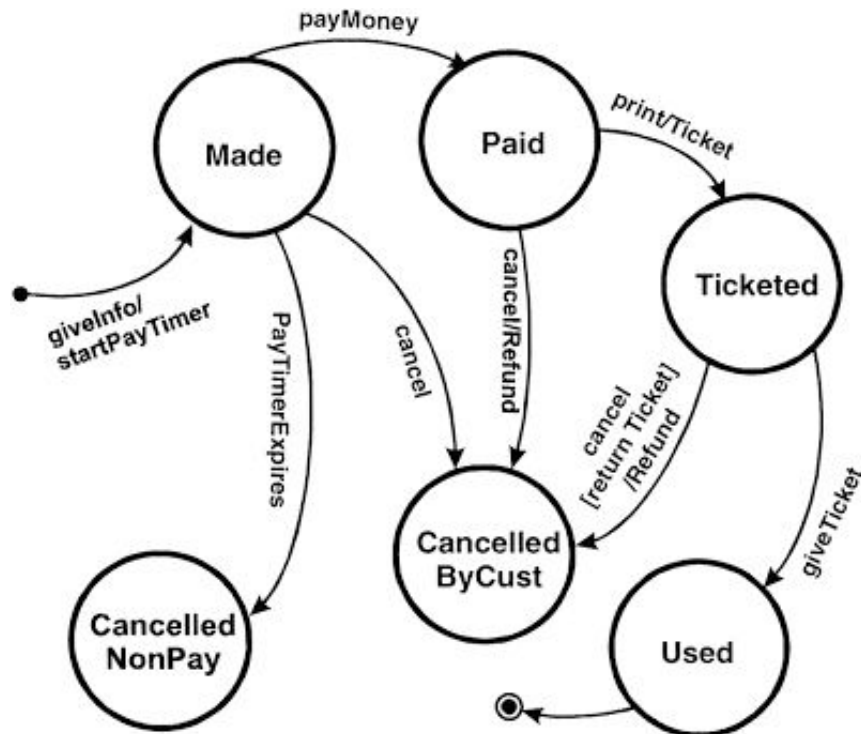
Можно протестировать одно число из каждого диапазона. Например: 5, 15, 20, 60. А также граничные значения (первое и последнее значения из каждого диапазона): 0, 13, 14, 17, 18, 54, 55, 99.

Пример со скидками

- Рисуем...

State transition testing

- Тестирование переходов из одного состояния системы, в другое



Причина/ Следствие (Cause/Effect)



Причина/ Следствие (Cause/Effect)

- Это ввод комбинаций условий (причин), для получения ответа от системы (следствие). Например, вы проверяете возможность добавлять клиента, используя определенную экранную форму. Для этого вам необходимо будет ввести несколько полей, таких как "Имя", "Адрес", "Номер Телефона" а затем, нажать кнопку "Добавить" - эта "Причина". После нажатия кнопки "Добавить", система добавляет клиента в базу данных и показывает его номер на экране - это "Следствие".

Причина/ Следствие (Cause/Effect)

- Пример с регистрацией нового пользователя.
- Что будет являться следствием ?

Предугадывание ошибок

- Это когда тест аналитик использует свои знания системы и способность к интерпретации спецификации на предмет того, чтобы "предугадать" при каких входных условиях система может выдать ошибку. Например, спецификация говорит: "пользователь должен ввести код". Тест аналитик, будет думать: "Что, если я не введу код?", "Что, если я введу неправильный код?", и так далее. Это и есть предугадывание ошибки.

Дымовое (Smoke testing).

Дымовое тестирование-это короткий цикл тестов, выполняемый для подтверждения того, что после сборки кода (нового или исправленного) устанавливаемое приложение, стартует и выполняет основные ф



Таблица принятия решений

- Вы хотите купить абонемент в тренажерный зал на 1 месяц. Ниже указаны условия:
- 1) В августе цена на 10% ниже
- 2) В январе цена на 10% выше
- 3) Если вы студент, то получаете 20% скидку
- 4) Если Вам больше 60 лет, то вы получаете 30% скидку (данная скидка не суммируется со студенческой)

Таблица принятия решений

Decision Table

Conditions	TC1	TC2	TC3	TC4
Request login	0	1	1	1
Valid user name entered	X	0	1	1
Valid password entered	X	X	0	1
Actions				
Offer recovery credentials	0	1	1	0
Activate entrybox user name	0	1	1	0
Activate entrybox password	0	0	1	0
Enter privileged area	0	0	0	1

Таблица принятия решений

- Попрактикуемся...

AD HOC

- Это тестирование без подробных спецификаций, сопроводительных документов, тест плана и т.д. Другими словами, тестирование в полном хаосе. Преимущество такой техники в том, что нет необходимости в планировании и документации, наиболее важные ошибки находятся быстро, нет задержек со стартом проекта.

Попарное тестирование (Pairwise testing).

Параметр 1	Параметр 2	Параметр 3
Значение 1.1	Значение 2.1	Значение 3.1
Значение 1.2	Значение 2.2	Значение 3.2

Переберем значения первого параметра со вторым (строки №1-4), первого с третьим (строки №5-8) и второго с третьим (строки №9-12). Удалив повторяющиеся наборы параметров (выделены серым), получим следующую таблицу тестов:

#	Параметр 1	Параметр 2	Параметр 3
1	Значение 1.1	Значение 2.1	Значение 3.1
2	Значение 1.1	Значение 2.2	Значение 3.1
3	Значение 1.2	Значение 2.1	Значение 3.1
4	Значение 1.2	Значение 2.2	Значение 3.1
5	Значение 1.1	Значение 2.1	Значение 3.2
6	Значение 1.1	Значение 2.1	Значение 3.2
7	Значение 1.2	Значение 2.1	Значение 3.1
8	Значение 1.2	Значение 2.1	Значение 3.2
9	Значение 1.1	Значение 2.1	Значение 3.1
10	Значение 1.1	Значение 2.1	Значение 3.2
11	Значение 1.1	Значение 2.2	Значение 3.1
12	Значение 1.1	Значение 2.2	Значение 3.2



#	Параметр 1	Параметр 2	Параметр 3
1	Значение 1.1	Значение 2.1	Значение 3.1
2	Значение 1.1	Значение 2.2	Значение 3.1
3	Значение 1.2	Значение 2.1	Значение 3.1
4	Значение 1.2	Значение 2.2	Значение 3.1
5	Значение 1.1	Значение 2.1	Значение 3.2
6	Значение 1.2	Значение 2.1	Значение 3.2
7	Значение 1.1	Значение 2.2	Значение 3.2

Зеленым выделены уникальные пары всех параметров в таблице. Теперь начинается самое интересное, значения выделенные белым не являются необходимыми для перебора всех пар в таблице, поэтому могут быть заменены на любое другое значение. Поэтому заменив их, мы можем оптимизировать тесты, добавив проверку пар из 5, 6 и 7 строк во вторую и третью строки, получим:

Параметр 1	Параметр 2	Параметр 3	
1	Значение 1.1	Значение 2.1	Значение 3.1
2	Значение 1.1	Значение 2.2	Значение 3.2
3	Значение 1.2	Значение 2.1	Значение 3.2
4	Значение 1.2	Значение 2.2	Значение 3.1



5	Значение 1.1	Значение 3.2
6	Значение 1.2	Значение 3.2
7	Значение 2.2	Значение 3.2

Как видно из примера выше, оптимизация даже такого малого набора параметров не так проста как могло бы показаться. При этом сложность задачи возрастает пропорционально росту числа параметров. Однако эта задача решаема, в чем мы убедимся в последствии.

All pairs testing

- Пример из жизни:
- Подобрать тестовые конфигурации на основе требований к поддерживаемым ОС, Версиям ОС, Браузерам, Разрешениям экрана

ВОПРОСЫ



Thank
You!