



Тестирование программного обеспечения

+7 (913) 768 8364

Ул. Кутателадзе 4г, к.118

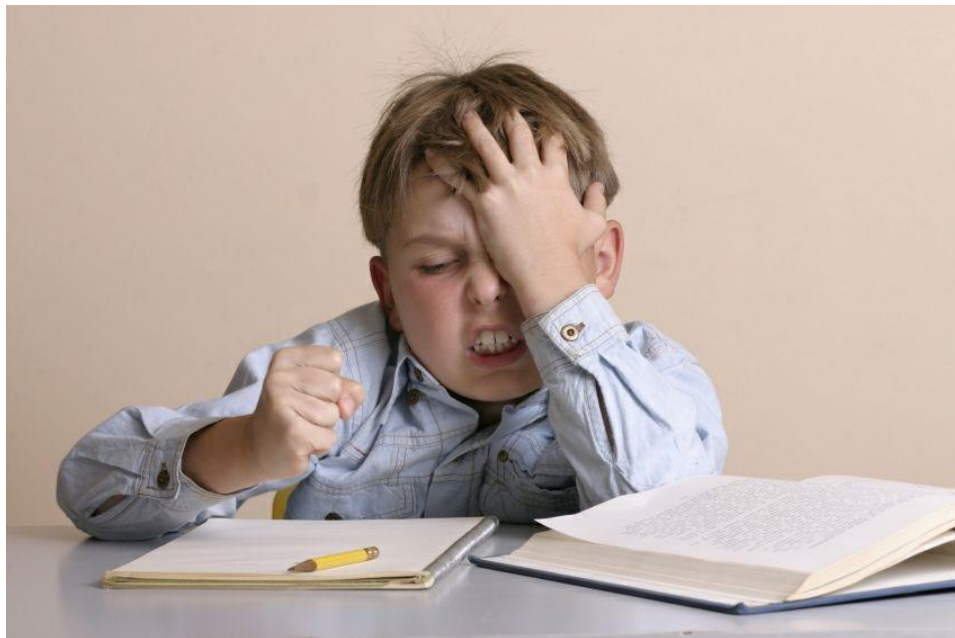
<https://academ-it-school.ru/>

Вопросы



Второе домашнее задание

- Как справились?
- Что изучали дополнительно?
- Есть ли вопросы?



План занятия

- Тестовая документация
 - План тестирования
 - Чек-листы
 - Тест-кейсы
 - Наборы тест-кейсов
- Матрица трассировки требований
- Отчеты
 - Отчет о дефекте
 - Отчет по результатам тестирования

Тест-кейс и поля тест-кейса?



Тест-кейс и поля тест-кейса

- **Тест-кейс (test case)** — набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки ПО
- **Поля тест-кейса**
 - Идентификатор (ID)
 - Приоритет (Priority)
 - Связанное требование (Related task)
 - Название (Summary)
 - Предусловия (Preconditions)
 - Шаги воспроизведения (Steps to reproduce)
 - Ожидаемый результат (Expected Result)

Заглавие (название)

- Информативность
- Хотя бы относительная уникальность (чтобы не путать разные тест-кейсы)
- Краткое и емкое, четко отражающее суть тест-кейса

Плохо	Хорошо
Тест 1	Запуск, одна копия, верные параметры
Тест 2	Запуск одной копии с неверными путями
Тест 78 (улучшенный)	Запуск, много копий, без конфликтов
Остановка	Остановка по Ctrl+C
Закрытие	Остановка закрытием консоли
...	...

Шаги воспроизведения

- Начинайте с понятного и очевидного места, но не пишите лишних начальных шагов
- Если пишете на русском языке, используйте безличную форму (например, «открыть», «ввести», «добавить» вместо «откройте», «введите», «добавьте»);
- Соотносите степень детализации шагов и их параметров с целью тест-кейса, его сложностью, уровнем и т.д.
- Пишите шаги последовательно, без условных конструкций вида «если... то...»

Ожидаемый результат

- Описывайте поведение системы так, чтобы исключить субъективное толкование
- Пишите ожидаемый результат по всем шагам, если у вас есть хоть малейшие сомнения в том, что результат некоего шага будет совершенно тривиальным и очевидным
- Пишите кратко, но не в ущерб информативности;
- Избегайте условных конструкций вида «если... то...».

Свойства качественных тест-кейсов

- Правильный технический язык
 - Не знаешь – используй Google
- Соответствие принятым шаблонам оформления
- Баланс между специфичностью и общностью
- Один тест-кейс – одна проверка (или один пользовательский сценарий)
- Последовательность в достижении цели
- Отсутствие лишних действий

Что почитать:

С. Куликов, «Тестирование ПО», глава 2.4.5

https://careers.epam.by/content/dam/epam/by/book_epam_by/Software_Testing_Basics_2_izdanie.pdf

Почему плоха излишняя специфичность?

- При повторных выполнениях тест-кейса всегда будут выполняться строго одни и те же действия со строго одними и теми же данными, что снижает вероятность обнаружения ошибки (эффект пестицида).
- Возрастает время написания, доработки и даже просто прочтения тест-кейса.

Почему плоха излишняя общность?

- Тест-кейс сложен для выполнения начинающими тестировщиками или даже опытными специалистами, недавно подключившимися к проекту.
- Тестировщик, выполняющий тест-кейс, может понять его иначе, чем было задумано автором.

Баланс между специфичностью и общностью

Конвертация из всех поддерживаемых кодировок

Шаги	Ожидаемые результаты
<p>Приготовления: Создать папки C:/A, C:/B, C:/C, C:/D. Разместить в папке C:/D файлы 1.html, 2.txt, 3.md из прилагаемого архива.</p>	
<p>1. Запустить приложение, выполнив команду «php converter.php c:/a c:/b c:/c/converter.log».</p> <p>2. Скопировать файлы 1.html, 2.txt, 3.md из папки C:/D в папку C:/A.</p> <p>3. Остановить приложение нажатием Ctrl+C.</p>	<p>1. Отображается консольный журнал приложения с сообщением «текущее_время started, source dir c:/a, destination dir c:/b, log file c:/c/converter.log», в папке C:/C появляется файл converter.log, в котором появляется запись «текущее_время started, source dir c:/a, destination dir c:/b, log file c:/c/converter.log».</p> <p>2. Файлы 1.html, 2.txt, 3.md появляются в папке C:/A, затем пропадают оттуда и появляются в папке C:/B. В консольном журнале и файле C:/C/converter.log появляются сообщения (записи) «текущее_время processing 1.html (KOI8-R)», «текущее_время processing 2.txt (CP-1251)», «текущее_время processing 3.md (CP-866)».</p> <p>3. В файле C:/C/converter.log появляется запись «текущее_время closed». Приложение завершает работу.</p>

Конвертация из всех поддерживаемых кодировок

Шаги	Ожидаемые результаты
Выполнить конвертацию трёх файлов допустимого размера в трёх разных кодировках всех трёх допустимых форматов.	Файлы перемещаются в папку-приёмник, кодировка всех файлов меняется на UTF-8.

Баланс между простотой и сложностью

- Преимущества простых тест-кейсов:
 - их можно быстро прочесть, легко понять и выполнить;
 - понятны начинающим тестировщикам и новым людям на проекте;
 - делают наличие ошибки очевидным;
 - упрощают начальную диагностику ошибки, т.к. сужают круг поиска.

Баланс между простотой и сложностью

- Преимущества сложных тест-кейсов:
 - при взаимодействии многих объектов повышается вероятность возникновения ошибки
 - пользователи, как правило, используют сложные сценарии, а потому сложные тесты более полноценно эмулируют работу пользователей
 - программисты редко проверяют такие сложные случаи

Как хорошо писать тест-кейсы

- Разделять систему на составляющие – делать декомпозицию – раскладывать ее на составные части
- Собирать и анализировать требования к продукту
- Верно расставлять приоритеты
- Формулировать свои мысли (письменно и устно) в понятном для других виде
- Знать и применять техники тест-дизайна

Алгоритм создания эффективных проверок

- Приступая к продумыванию чек-листа, тест-кейса или набора тест-кейсов, задайте следующие вопросы:
 - Что перед вами?
 - Кому и зачем оно нужно (и насколько это важно)?
 - Как оно обычно используется?
 - Как оно может сломаться, т.е. начать работать неверно?

Типичные ошибки

- Отсутствие заглавия или плохо написанное, неинформативное заглавие
- Постоянное использование слов «проверить», «протестировать» (и подобных)

Плохо	Хорошо
Проверить запуск приложения. Проверить открытие корректного файла. Проверить модификацию файла. Проверить сохранение файла. Проверить закрытие приложения.	Запуск приложения. Открытие корректного файла. Модификация файла. Сохранение файла. Закрытие приложения.

- Описание стандартных элементов интерфейса своими словами вместо использования их устоявшихся названий

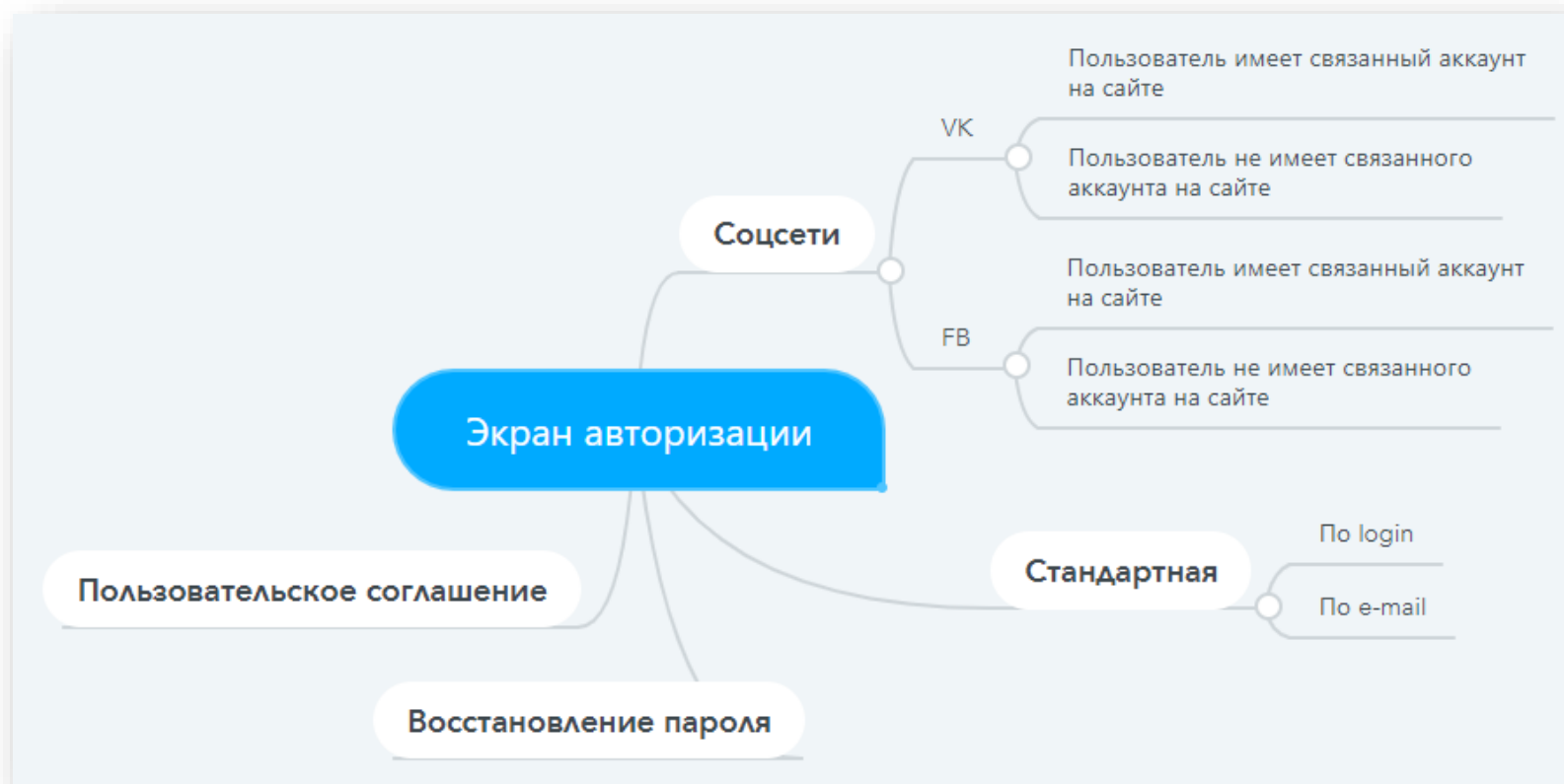
Типичные ошибки

- Ссылка на другие тест-кейсы или шаги других тест-кейсов
- Расплывчатые двусмысленные описания действий и ожидаемых результатов
- Отсутствие описания приготовления к выполнению тест-кейса

Шаги выполнения	Ожидаемые результаты
<ol style="list-style-type: none">1. Нажать на панели «Главная» кнопку «Быстрая дедубликация».2. Выбрать каталог «C:/MyData».	<ol style="list-style-type: none">1. Кнопка «Быстрая дедубликация» переходит в утопленное состояние и меняет цвет с серого на зелёный.2. На панели «Состояние» в поле «Дубликаты» отображается «~200».

Системы хранения тестов

- MS Office, Google Docs
- Интеллект-карты (mind maps)



Системы хранения тестов

- JIRA (Zephyr)

The screenshot displays the JIRA Zephyr interface for a test cycle titled "Verify that the Appstore can be accessed". The interface is divided into several sections:

- Header:** JIRA logo, navigation tabs (Dashboards, Projects, Issues, Tests), a "Create" button, and a search bar.
- Test Cycle Header:** "FIRE / FIRE-14" and the test cycle title.
- Actions:** Buttons for "Comment", "Attach Files", "More", "Execute", "Add to Test Cycle(s)", and "Export".
- Details:**
 - Type: Test
 - Priority: Medium
 - Affects Version/s: None
 - Labels: appstore
 - Status: OPEN (View Workflow)
 - Resolution: Unresolved
 - Fix Version/s: None
- People:**
 - Assignee: Unassigned
 - Reporter: Lana Malakova
 - Votes: 0
 - Watchers: Stop watching this issue
- Dates:**
 - Created: 7 minutes ago
 - Updated: 7 minutes ago
- Description:** Appstore specific tests. Needs integration with the latest build of appstore platform.
- Test Details:**

Test Step	Test Data	Test Result
1 Turn on phone. From the main screen locate the appstore icon.		Appstore icon should be present
2 Click/Tap on the icon	Single tap	The app should open
- Test Executions:**

Version	Test Cycle	Status	Defects	Executed By	Executed On	
Release 1.0	Mobile Testing	FAIL	10 FIRE-15	lana.malakova	02-01-2015 18:14:15	E

Системы хранения тестов

- JIRA Cloud ([Issue Checklist Free](#))

Issue Checklist

Completion: 0 / 2

Add ToDo item or header text here...

☐ An item with description

Hide description ▼

This is description for the item above.

☐ An item without description

Системы хранения тестов

- TestLink

The screenshot displays the 'Create Test Case' interface in TestLink. The form is organized into several sections: a top header bar, a 'Test Case Title' field with a 'Create' button, a 'Summary' section with a rich text editor, and two side-by-side sections for 'Steps' and 'Expected Results', each also featuring a rich text editor. At the bottom, there is a 'Keywords' section with two list boxes for 'Available Keywords' and 'Assigned Keywords', connected by a set of arrows. A final 'Create' button is located at the bottom right of the form.

Create Test Case

Test Case Title

Summary

Font Size

Steps

Font Size

Expected Results

Font Size

Keywords




Available Keywords

Assigned Keywords

Матрица трассировки требований (traceability matrix)

Веб-приложение N									
Файл Правка Вид Вставка Формат Данные Инструменты Дополнения Справка Все изменения на Диске сохранены									
p. % .0 .00 123 Arial 10 B I A									
fx									
A B C D E F G H I									
Статусы Ссылки									
№ Раздел Функционал Приоритет Тех.Задание Тест-анализ Разр-ка Тех.Задание Тест-анализ									
1 Пользователь									
1.1 Создание нового пользователя									
Регистрация пользователя									
Создание нового пользователя без активации									
Создание нового пользователя с активацией									
Высокий Неактуально В процессе Готово https://docs.goc https://docs.goc									
1.2 Профиль пользователя									
Просмотр									
Редактирование основных данных									
Редактирование подписок									
Редактирование интеграции с BTS									
Высокий Готово В процессе Готово https://docs.goc https://docs.goc									
1.3 Авторизация									
Авторизация пользователя с разными статусами									
Высокий Готово Готово Готово https://docs.goc https://docs.goc									
2.1 Чек-лист (как список тестов)									
2.1.1 Создание нового чек-листа (как список тестов)									
Создание тестов вручную									
Создание тестов через импорт									
Критичный Готово В процессе Готово https://docs.goc https://docs.goc									
2.1.2 Формирование данных о прохождении чек-листа (Запуск)									
Запуск созданного чек-листа и его прохождения									
Высокий Готово В процессе Готово https://docs.goc https://docs.goc									
2.1.2 Редактирование чек-листа									
Внесение изменений в существующий чек-лист									
Высокий Неактуально Неактуально Неактуально https://docs.goc https://docs.goc									
2.1.2 Удаление чек-листа									
Удаление существующего чек-листа									
Высокий Готово Готово Готово https://docs.goc https://docs.goc									

Матрица трассировки требований (traceability matrix)

#	Type	Task	Planned date	Status	Comments/Subtask	Jira	PRI 1 MH 1.5 SH 2 NTH	Est
5600	Bug	Mobile adaptation After subscribing				 -158 - Mobile adaptation After subscribing CLOSED	2	
6200	Task	Blog article: add Author, Stay in the loop blocks and Navigation				 -172 - Blog article: add Author, Stay in the loop blocks and Navigation IN PROGRESS	2	
6300	Task	New Services Page				doing this	1	
6400	Task	revise services page				doing this	2	
6500	Task	update all Course & Tutorial XMLs and regenerate PDFs			<input type="checkbox"/> _____ check S3 for updated files and estimate effort. Let _____ know if we need her assistance.		1	1
6510	Task	Sync Courses and Tutorials data with the current prod				 -206 - Sync Courses and Tutorials data with the current prod RESOLVED	1	?

Практика

- Выбрать два основных пользовательских сценария для сайта [https://ru.wikipedia.org/wiki/Заглавная страница](https://ru.wikipedia.org/wiki/Заглавная_страница)
- Написать для каждого позитивный тест-кейс и негативный тест-кейс
- Классифицировать по видам тестирования

Окружение для тестирования, или тестовые комплексы

- Локально (local)
- Дев (dev)
- Тест (test)
- Стейдж (staging), Модель, Демо
- Прод (production), пром, бой

Что почитать:

https://ru.wikipedia.org/wiki/Окружения_развёртывания_программного_обеспечения

Вспомним

- Дефект — ?

Вспомним

- Дефект — расхождение ожидаемого и фактического результата.
- Ожидаемый результат — ?

Вспомним

- **Дефект** — расхождение ожидаемого и фактического результата.
- **Ожидаемый результат** — поведение системы, описанное в требованиях.
- **Фактический результат** — ?

Вспомним

- **Дефект** — расхождение ожидаемого и фактического результата.
- **Ожидаемый результат** — поведение системы, описанное в требованиях.
- **Фактический результат** — поведение системы, наблюдаемое в процессе тестирования.

Отчет о дефекте (баг-репорт)

- Отчёт о дефекте (defect report, bug report) — документ, описывающий ситуацию или последовательность действий, которая привела к некорректной работе объекта тестирования, с указанием причин и ожидаемого результата.

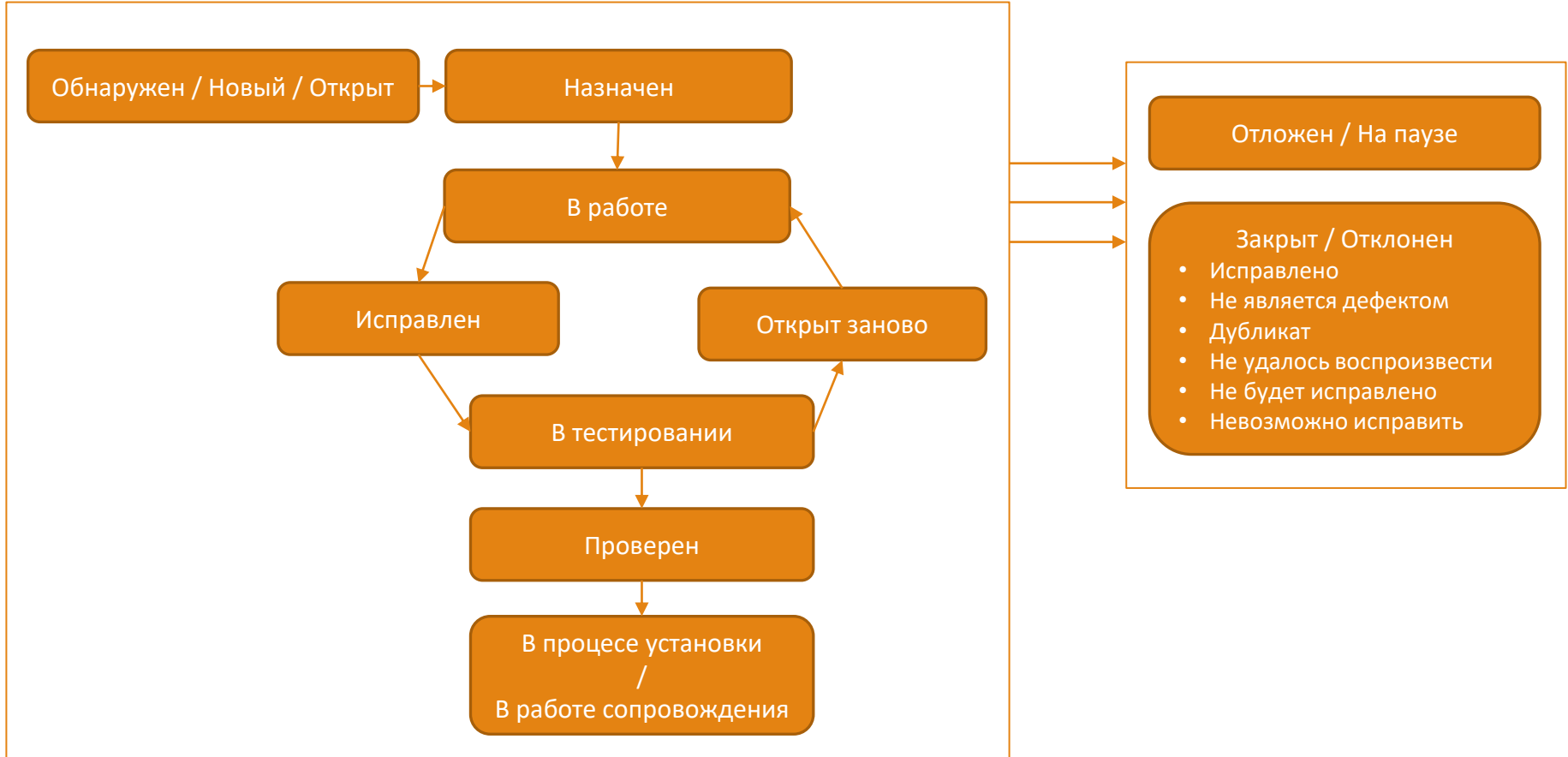
Что почитать:

<https://geteasyqa.com/ru/qa/write-bug-report/>

Для чего нужен отчет о дефекте?

- Предоставить информацию о проблеме
- Приоритизировать проблему
- Содействовать устранению проблемы

Стадии жизненного цикла отчёта о дефекте



Атрибуты отчета о дефекте

Поле	Описание
Идентификатор (ID)	Должен быть уникальным (т.е. не повторяться)
Краткое описание, заглавие (summary)	Что? Где? При каких условиях?
Подробное описание (description)	<ul style="list-style-type: none">• Подробное описание проблемы• Подготовительные действия• Шаги воспроизведения*• Ожидаемый результат*• Фактический результат*• Дополнительная информация
Воспроизводимость	Удастся ли вызвать дефект при каждом воспроизведении
Приоритет	Важность и срочность дефекта
Приложения	Любые вложения для более наглядного описания <ul style="list-style-type: none">• Скриншоты• Скринкасты• Логи

Краткое описание

- Поле, в котором нужно поместить весь смысл всего баг репорта
- Должно содержать предельно краткую, но в то же время достаточную для понимания сути проблемы информацию о баге
- Отвечать на три вопроса:
 - Что?
 - Где?
 - При каких условиях?
- Должно быть законченным предложением, построенным в соответствии с правилами грамматики

Краткое описание

Для создания хороших описаний дефектов рекомендуется пользоваться следующим алгоритмом:

1. Полноценно понять суть проблемы.
2. Сформулировать подробное описание (description) дефекта — сначала без оглядки на длину получившегося текста.
3. Выделить в подробном описании слова (словосочетания, фрагменты фраз), отвечающие на вопросы, «что, где и при каких условиях случилось».
4. Убрать из получившегося подробного описания всё лишнее, уточнить важные детали.
5. Оформить получившееся в пункте 4 в виде законченного грамматически правильного предложения.
6. Если предложение получилось слишком длинным, переформулировать его, сократив длину (за счёт подбора синонимов, использования общепринятых аббревиатур и сокращений).

Подробное описание

- Подробное описание проблемы
 - Используйте простой естественный язык
- Подготовительные действия (не обязательно)
 - Описание действий, которые необходимо предварительно выполнить или учесть
- Шаги воспроизведения
 - Как воспроизвести эту ошибку – шаг за шагом
- Ожидаемый результат
- Фактический результат
- Дополнительная информация (не обязательно)
 - Информация, которая считается важной, и которая поможет оперативно исправить дефект
 - Например, окружение, отсутствие сети, данные для входа и т.д.

Пример

- Подробное описание: если в систему входит администратор, на странице приветствия отсутствует логотип.
- Фактический результат: логотип отсутствует в левом верхнем углу страницы.
- Ожидаемый результат: логотип отображается в левом верхнем углу страницы.

Приоритет

- Важность – показывает степень ущерба, который наносится проекту существованием дефекта
- Срочность – показывает, как быстро дефект должен быть устранён
- Приоритет
 - Blocker – баг блокирует дальнейшую работу приложения или процесс тестирования
 - Critical – присваивается при значительном влиянии проблемы на поведение ПО (падение приложения, потеря данных, утечки памяти)
 - Major – присваивается при значительном влиянии проблемы на поведение ПО, когда работа ПО сильно отличается от эталонного
 - Minor – указывается при незначительном влиянии на эталонное поведение ПО
 - Trivial – баг не оказывает влияния на функционал и поведение ПО

Логика создания эффективных отчётов о дефектах

При создании отчёта о дефекте рекомендуется следовать следующему алгоритму:

1. Понять суть проблемы.
2. Сформулировать суть проблемы в виде «что сделали, что получили, что ожидали получить».
3. Заполнить поля отчёта, начиная с подробного описания.
4. После заполнения всех полей внимательно перечитать отчёт, исправив неточности и добавив подробности.
5. Ещё раз перечитать отчёт, т.к. в пункте 4 вы точно что-то упустили.

Типичные ошибки при написании отчётов о дефектах

- Ошибки формулировок
 - Плохие краткие описания
 - Идентичные краткое и подробное описания*
 - Отсутствие в подробном описании явного указания фактического результата и ожидаемого результата

Что почитать:

- <http://getbug.ru/3-plohih-privyichki-pri-sostavlenii-bag-reporta>
- <https://geteasyqa.com/ru/qa/write-bug-report/>

Типичные ошибки при написании отчётов о дефектах

- Ошибки оформления
 - Копии экрана в виде «копий всего экрана целиком»
 - Копии экрана, на которых не отмечена проблема

Типичные ошибки при написании отчётов о дефектах

- Логические ошибки
 - Выдуманные дефекты либо отнесение расширенных возможностей приложения к дефектам
 - Чрезмерно заниженные (или завышенные) важность и срочность
 - Субъективное мнение без указания аргументов
- Проверки
 - Не является ли баг дубликатом уже заведенного в баг-трекере?
 - Возможно ли предложить улучшение?

Пример

- Тестированию подвергается некое веб-приложение, поле описания товара должно допускать ввод максимум 250 символов; в процессе тестирования оказалось, что этого ограничения нет.

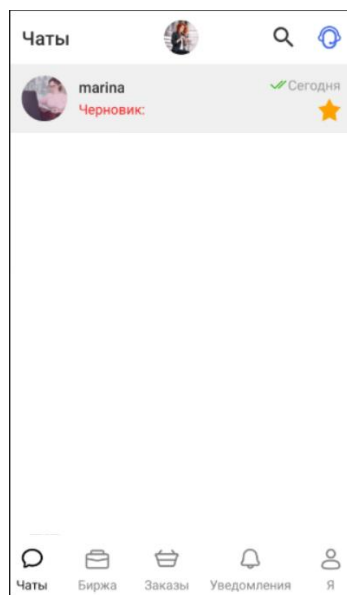
Пример

1. Суть проблемы: исследование показало, что ни на клиентской, ни на серверной части нет никаких механизмов, проверяющих и/или ограничивающих длину введённых в поле описания товара данных.
2. Исходный вариант подробного описания: в клиентской и серверной части приложения отсутствуют проверка и ограничение длины данных, вводимых в поле «О товаре» на странице <http://testapplication/admin/goods/edit/>.
3. Определение «что, где и при каких условиях случилось»:
 - Что: отсутствуют проверка и ограничение длины вводимого текста.
 - Где: описание товара, поле «О товаре», <http://testapplication/admin/goods/edit/>.
 - При каких условиях: — (в данном случае дефект присутствует всегда, вне зависимости от каких бы то ни было особых условий).
4. Сокращение (итоговое краткое описание): нет ограничения максимальной длины поля «О товаре».

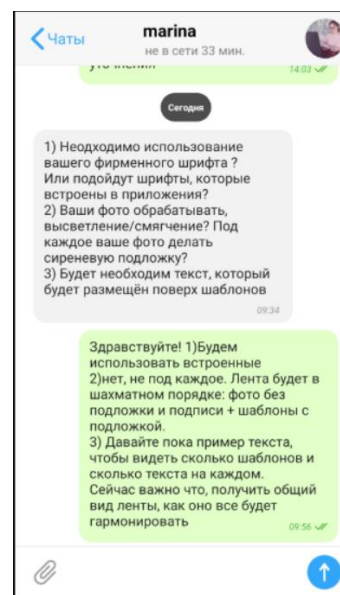
Практика

- Мобильное приложение-мессенджер. С жалобой на зависший черновик обратился пользователь:
 - Баг в мобильном приложении - сообщение зависло в «Черновике».
- Параметры Huawei P10. EMUI 9.1.0. Версия приложения 1.5.4.3. Аккаунт marina

- Скриншоты:



Висит надпись черновик



Нет черновика

- В десктопе - нет черновика.

Инструментальные средства управления отчётами о дефектах

- [Jira Cloud](#) (логин — mail.for.testbase@yandex.ru, пароль — 12345678) ©
- [Bugzilla](#) (логин — mail.for.testbase@yandex.ru, пароль — 12345678) ©
- [Youtrack](#) (логин — mail.for.testbase@yandex.ru, пароль — 12345678) ©
- [Redmine](#) (пользователь — Test, пароль — 12345678) ©
- Trello
- Assembla

© Данные взяты из блога <http://okiseleva.blogspot.ru/>

Отчет по результатам тестирования

- Документ, содержащий информацию о выполненных действиях (протестированные модули, используемые тест-кейсы, примененные виды тестирования и т.д.) и результаты этой работы (соотношение пройденных и непройденных тест-кейсов, количество и приоритет ошибок, затраченное время и т.д.).
- В общем виде:
 - Что тестировали
 - Как тестировали
 - Какие выводы можете сделать

Отчет по результатам тестирования

- Подробно:
 - Название и версия ПО
 - Если не прилагается тест-план
 - Какие модули были протестированы
 - Как именно модули были протестированы
 - Количество используемых тест-кейсов \ чек-листов
 - Соотношение пройденных и непройденных проверок
 - Общее количество обнаруженных ошибок
 - Приоритеты ошибок
 - Общие итоги
 - Насколько непройденные тест-кейсы и обнаруженные ошибки влияют на функционал
 - Возможен ли выпуск продукта по результатам тестирования
 - Какие рекомендации по улучшению можно дать

Домашнее задание 3

- Задание 1
 - Написать план тестирования сайта <https://tritonshoes.ru/>
 - В тест-план включить тест-кейсы – ровно 10 с учетом приоритетов при ограниченном времени на тестирование
 - Для каждого тест-кейса:
 - Указать приоритет
 - Классифицировать по позитивности сценариев
 - В тест-план включить чек-лист для UI/UX
 - 5 проверок
 - Написать отчет по результатам тестирования
- Задание 2
 - Оформить 4 баг-репорта по найденным (либо предполагаемым) багам в баг-трекере JIRA
 - Прислать скриншоты либо выгрузку баг-репортов

Что еще поизучать

- Разобрать примеры тест-кейсов
 - <http://okiseleva.blogspot.com/2014/08/blog-post.html>
- Потренироваться в чек-листах и тест-кейсах – бесплатная часть Яндекс.Практикум по тестированию
 - <https://praktikum.yandex.ru/profile/qa-engineer/>