Артефакты тестирования. Дефекты.

- артефакты документально подтверждают проделанную тестировщиком работу.
- помогают участникам разработки не путаться в проблемах и новых фичах
- помогают тестировщикам собирать статистику о своей работе: о скорости ,качестве тестирования и так далее.

Что такое артефакты тестирования ?

Выделяют следующие артефакты:

1. **Баг-репорт**
2. **Отчет о тестировании**
3. Тест план
4. Тест кейс
5. Тест сьют
6. Чек-лист

Что важно при создании артефактов?

- Грамотный и понятный русский язык;
- Писать тесты так, чтобы человек ,который их видит впервые, все понял;
- Не забывать о тестовых учетках и подробном описании тестового окружения,нужного для прохождения;
- Умение декомпозировать(разбивать большое трудное действие на несколько простых действий) сервис на компоненты: например, настройки,авторизация и так далее.
- Расстановка приоритетов :например,проверить оплату и авторизацию обычно важнее , чем смену фоновой картини;
- Умение применить нужные техники тест дизайна.

Баг репорт

Баг или дефект-репорт - это документ ,описывающий ситуацию или последовательность действий ,приведшую к некорректной работе объекта тестирования , с указанием причин и ожидаемого результата.

Главное , что нужно запомнить - название и описание должны быть понятны ЛЮБОМУ участнику команды ,от менеджера до сотрудника саппорта(тех.поддержки).

Системы баг-трекинга

Оформление багов

По названию бага мы должны сразу понимать ,где и что произошло.

В содержании **обязательно** должны быть:

- Предусловие воспроизведения (если он есть);(что привело к этому)
- Последовательность действий для воспроизведния;
- Фактический результат;
- Ожидаемый результат.

Важная информация в описании бага :

- Окружение/условие воспроизведения;
- Снимки экрана/видео;
- Логи/артефакты по работе ПО;
- Атрибуты ошибки(важность,компонент и т.д.)

Как правильно назвать баг-репорт?

Простой способ кратко,но понятно назвать баг-репорт -**последовательный ответ на три вопроса**:

- **ЧТО** не так работает?
- **ГДЕ** не работает , в каком месте продукта?
- **КОГДА** данная проблема проявляется?

Баг-репорт : градация серьезности *(знать)*

Критичность багов

S1 – Блокирующий (Blocker) – дефект полностью блокирует выполнение функционала, нет никакого способа его обойти.

S2 – Критический (Critical) – дефект блокирует часть функциональности, но есть альтернативный путь для его обхода.

S3 – Значительный (Major) – дефект, указывающий на некорректную работу части функциональности. Зачастую связан не с тем, что функция не работает, а с тем, что она работает неправильно.

S4 – Незначительный (Minor) – дефект, не относящийся к функциональности системы. Обычно серьезность Minor проставляется для тех дефектов, которые относятся к удобству использования или интерфейсу.

Градация приоритетности*(знать)*

Ставят менеджеры , статус того, насколько быстро нужно справиться с задачей

Астрал
Академия

#5. Отчёты о дефектах

Приоритет багов

P1 –Высокий (High) – ошибка должна быть исправлена как можно быстрее, т.к. ее наличие является критической для проекта.

P2 – Средний(Medium) – ошибка должна быть исправлена, ее наличие не является критичной, но требует обязательного решения.

P3 – Низкий (Low) – ошибка должна быть исправлена, ее наличие не является критичной, и не требует срочного решения.

P4 – Минимальный (Min) – ошибка не обязательна к исправлению, так как текущий функционал не актуален или в ближайшее время будет изменен

Определяется менеджером (PM/PO) проекта на основании влияния проблемы на бизнес задачи/цели программного продукта.

Градация частотности

Градация того насколько часто производится ,повторяется баг.

Определяется специалистом по маркетингу или специалистом по качеству проекта на основании анализа поведенческих паттернов конечных пользователей или других данных по конкретному проекту.Есть не на всех проектах, но все равно важно ее осознавать.

- Высокая(High)/Always
- Средняя(Medium)/Often
- Низкая(Low)/Sometimes
- Незначительная (Very low)/Once

Признаки отличного баг-репорта

- Воспроизводимость
- Наличие обязательных пунктов, описанных ранее
- Уникальный номер и название
- Отсутствие агрессии и командного тона в описании
- Четкость интерпретации
- Логи и медиаматериалы хорошего качества
- Серьезность, соответствующая проблеме
- Используйте верную терминологию

BUG SEVERITY STANDARDS

	GUI	Functional
Critical		• Crash Error • Undercharging or Overcharging Customers • Code Issue
High	• Error Message Display Issue	• 404 Errors / Redirection Issue • Connectivity Issue • Social Integration error • Missing features/buttons
Medium	• Layout Issue	• Validation Issue
Low	• Font Issue • Colour Issue	• Hyperlink Issue
Trivial	• Tooltip Issue	

Баги можно разделить на типы, рассмотрим самые часто встречающиеся:

- функциональные
- логические
- визуальные
- ошибки контента
- ошибки удобства использования

Зачем нужно разделение?

Для структуризации информации для себя и для статистики ,например!

Функциональные

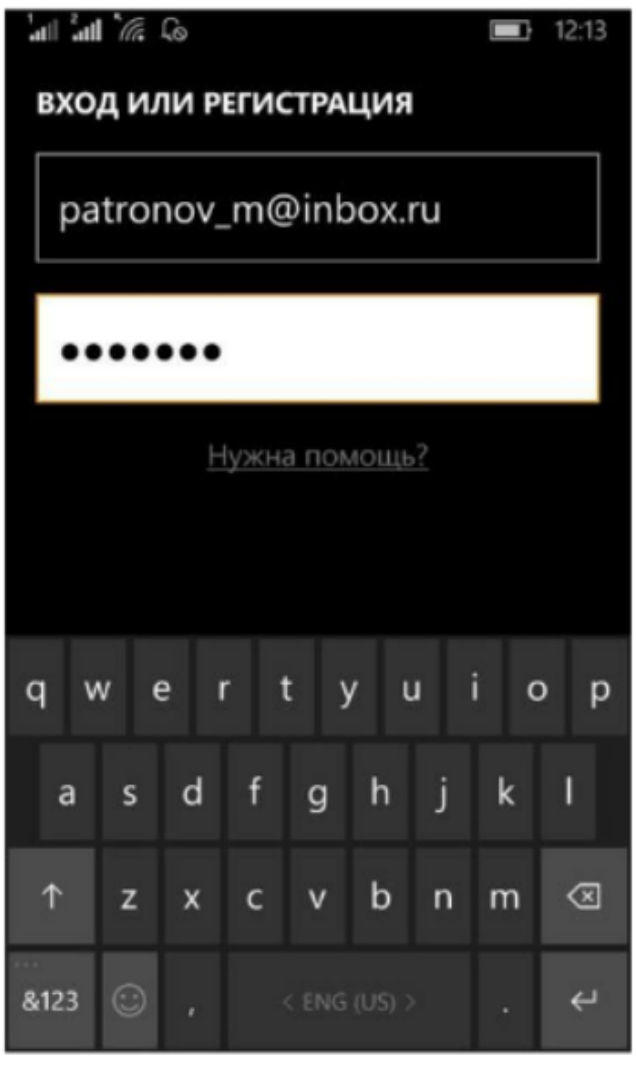
баги которые начинаются со слов : не работает

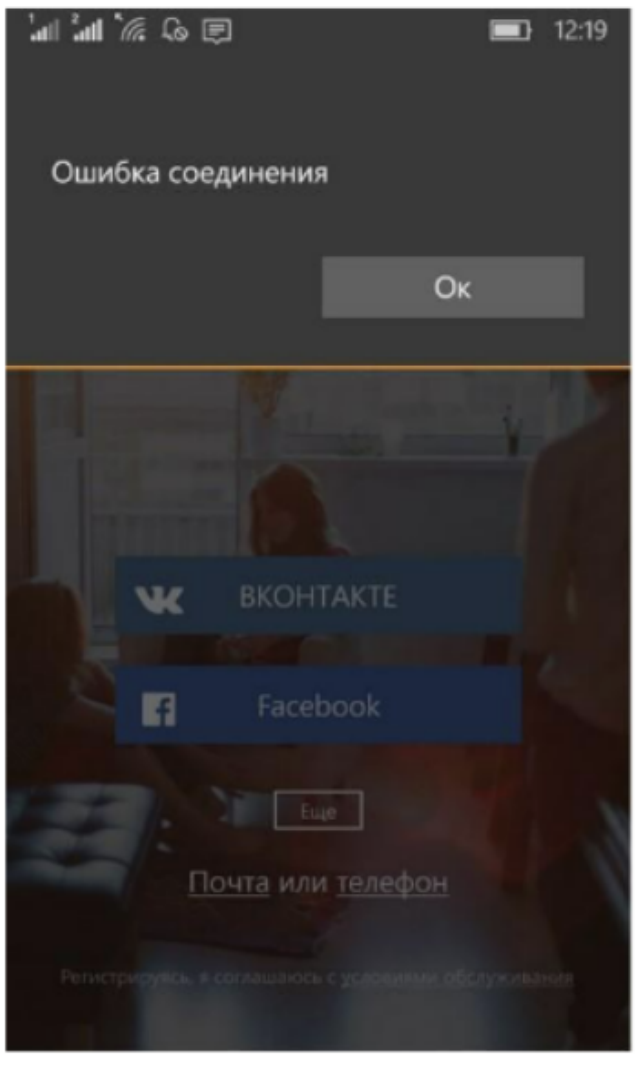
Пример:

- не работает шаринг(кнопка поделиться) в инстаграм
- не добавляются товары в корзину

- не работает авторизация

Функциональные





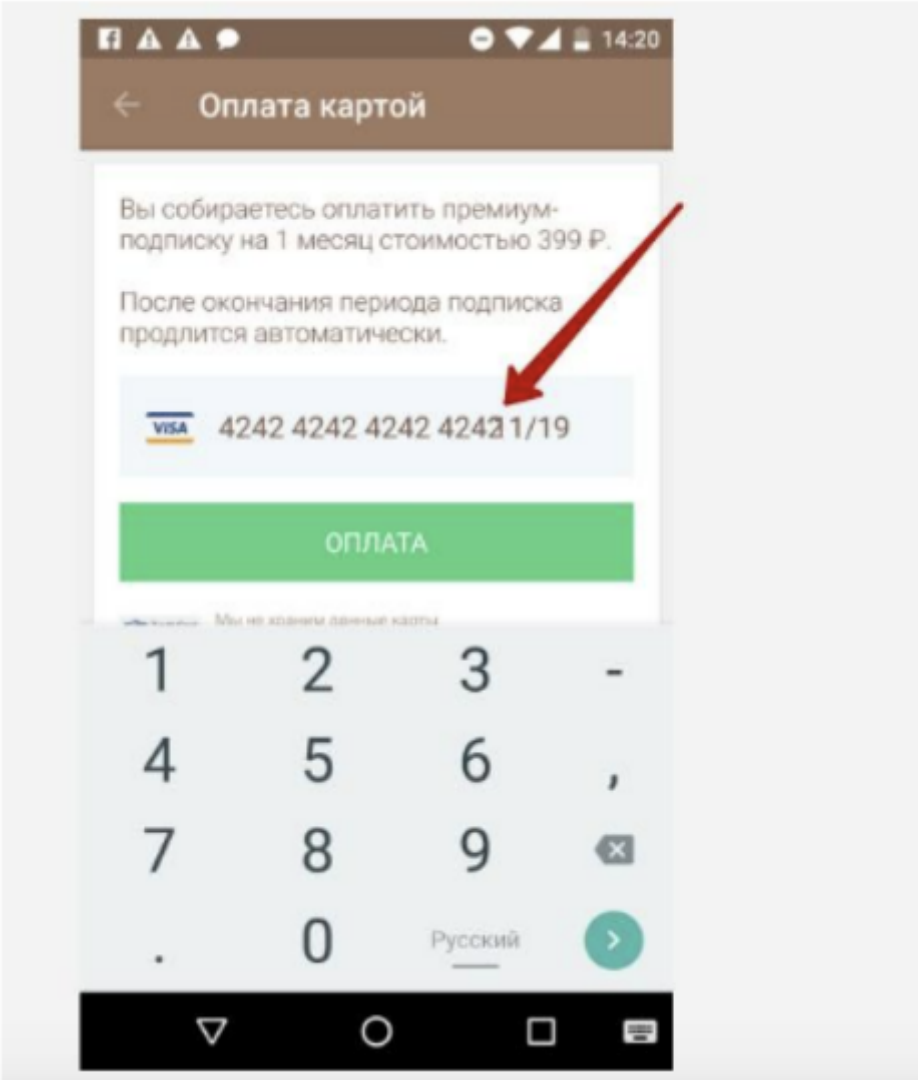
Логические

- можно перейти на оплату товара,если корзина пустая
- можно ввести дату 404 апреля
- неверная логическая последовательность

Визуальные

Примеры:

- Битая картинка
- Наезжает текст
- Неправильный шрифт



Контент

Примеры:

- Несоответствие контента друг другу
- Неправильный курс валют

Удобство использования

Примеры:

- Сброс значений при неверном вводе
- Перегруженный или слишком минималистичный интерфейс
- Проблемы с tap area

Какие могут быть еще ?

- локализации
- безопасности
- системные
- взаимодействие с железом
- связанные со стандартами
- и прочие..

Локализация дефекта

Вопрос который нужно держать в голове : Как еще повторить этот баг?

Где ,как,почему и при каких условиях баг точно повторяется?

Когда собрана общая информация о системе , мы можем локализировать баг и собрать необходимое максимальное количесвто информации о его воспроизведении:

- **Выявить причиный возникновения дефекта**

Например,не проходит восстановление пароля. Необходимо выявить,откуда приходит запрос на сервер в неверном формате - от backend либо frontend . Эти данные можно получить из логов.

- **Проанализировать возможность влияния найденного дефекта на другие области**

Например , в одной из форм ,которую редко используют,возникает ошибка при нажатии на кнопку “Редактировать”.

- **Отклонение от ожидаемого результата**

Нужно проверить, соответствует ли результат тестирования ожидаемому результату.Справится с этой задачей помогает техническое задание.

- **Исследовать окружение**

Необходимо воспроизвести баг в разных операционных системах (iOS,Android,Windows и т.д.) и браузерах (Google Chrome ,Mozila,Internet Explorer и др.).

При этом нужно проверить требования к продукту,чтобы выяснить,какие системы должны поддерживаться.

- **Проверить на разных устройствах**

Например ,desktop-приложение предназначено для использования на компьютерах,поэтому зачастую нет необходимости тестировать его на мобильных устройствах.

Для смортфонов в идеале должна быть разработана отдельная мобильная версия, которую , в свою очередь,нет смысла тестировать на компьютерах. Кроме того ,есть web приложения, которые могут работать и на компьютерах , и на мобильных устройствах , и тестировать их нужно на всех типах устройств.

- **Проверить в разных версиях ПО**

Для того чтобы не запутаться в реализованных задачах, в разработке используют версионность ПО.

- **Проанализировать ресурсы системы**

Возможно,дефект был найден при нехватке внутренней или оперативной памяти устройства. В таком случае баг может воспроизводиться на идентичных устройствах по-разному.

Отчет о тестировании

Отчет о тестировании - это документ , содержащий информацию о выполненных действиях ,результатах проведенной работы . Обычно он включает в себя таблицы,графики,списки,просто описывающую информацию в виде текста.

Перед тем как написать его ,нам надо понять ,что мы хотим в нем описать и для кого.

Когда нам нужен отчет? :

- При завершении регрессионного тестирования;
- При завершении исследовательского тесстирования нового проекта;
- При промежуточной отчетности при реализации новой фичи.

Схема того,что может быть в отчете:

Тестовая документация

Помимо описанного выше, в тестовую документацию обычно входит:

- Техническое задание;
- Частное техническое задание;
- База знаний (напрмиер , в Wiki);
- Медиа - материалы.

Техническое задание

ТЗ(техническое заданеи) - позволяет донести суть предмета разработки до сотрудников компании. Помогает понять,какой именно функиональностью должен обладать разрабатываемый продукт.

- Все сотрудники могут узнать про то ,как работает проект и своевременно узнавать об изменениях;
- Быстрое введение в курс дела новых сотрудников;
- Помощь тестированию,например,для написания положительных сценариев;
- И многое другое.

Тестирование требований

Тестирование требований направлено на то , чтобы уже на начальных этапах проектирования системы устранить максимальное возможное количество ошибок . Результаты:

- найти и исправить баг в требованиях дешевле чем найти баг уже после релиза ,от этого зависит конечная стоимость проекта;
- лучшее качесвто продукта;
- сохранение нервных клеток).

Пармаетры требований

Параметры оценивания :

- актуальность
- четкость и ясность
- учтено максимальное количество сценариев(и негативных тоже!)
- логичность
- выполнимость
- проверяемость

**Лог-это запись того что происходило в приложении*