ПРАКТИЧЕСКОЕ ЗАДАНИЕ №10 БАГ-РЕПОРТЕР

Цель: ознакомиться с понятием баг-репорт.
Время выполнения: 90 минут (1 пара)

ТЕОРИЯ

Система отслеживания ошибок (от англ. «bug tracking system») – прикладная программа, которая разработана, чтобы помочь программистам и тестировщикам контролировать ошибки и неполадки, найденные в программах. Кроме того, система помогает отслеживать, учтены ли пожелания пользователей и устранены ли ошибки.

Баг-трекер – хранилище истории изменений в ПО. Каждый из членов команды разработки использует баг-трекер для своих целей:

- **Руководитель проекта** – следит за выполнением задач, количеством ошибок и решает, что и когда будет исправляться. Часть обязанностей он может делегировать;
- **Аналитик** – формулирует задачи: требования, описание сценариев использования функциональности;
- **Разработчик** – использует баг-трекер как список дел. Он реализует функциональность, описанную в порученной ему задаче, или исправляет ошибку, которую ему назначили;
- **Тестировщик** – проверяет реализованные разработчиком задачи, регистрирует дефекты, которые обнаружены в ПО или получены от пользователей.

Баг-трекер отображает текущее состояние ПО: как много разработчики уже реализовали, какие ошибки есть в системе, какая доля из них исправлена и сколько еще нужно разработать. В некоторых баг-трекерах есть подобие новостной ленты, в которой отражена вся работа команды: кто и какие изменения внес, какие новые ошибки найдены, кто и кому переназначил задачи.

Важная функция баг-трекера – оповещение заинтересованных лиц в изменениях по задачам и ошибкам. Чтобы не бегать к разработчику и не сообщать, что в ПО нашлась ошибка, ее записывают и при необходимости назначают на разработчика. При следующем обновлении «списка дел» он уже будет в курсе проблемы. Как только разработчик закончил реализацию функциональности по задаче или исправил ошибку, то он назначает ее тестировщику. Это станет для него оповещением, что можно приступать к проверке.

Хороший баг-репорт позволяет:
1. воспроизвести проблему (это не всегда возможно, но надо стремиться).
2. понять, в чем проблема и какова ее важность.

Как написать хороший баг-репорт?

Для начала надо подготовиться. Если вы обнаружили баг, не стоит моментально бежать в баг-трекер и писать «ничего не работает!». Воспроизведите ошибку. Воспроизвелась? Отлично. Не воспроизвелась? Значит, что-то вы не учли. Вспоминайте, что делали.

Обязательные поля

НАЗВАНИЕ ПОЛЯ	ОПРЕДЕЛЕНИЕ
ID	Уникальный номер баг-репорта.
Заголовок	Суть ошибки.
Шаги воспроизведения	Алгоритм, пошаговая инструкция, как воспроизвести баг.
Результаты	Описание ФР и ОР.
Окружение	Операционные системы, браузеры или версии приложений, в которых возникает ошибка.
Приоритет	Критичность ошибки и срочность её исправления.

Необязательные поля

НАЗВАНИЕ ПОЛЯ	КОГДА ДОБАВЛЯТЬ
Описание	Создают, если в заголовке недостаточно информации о дефекте.
Предусловие	Указывают, когда систему нужно подготовить перед тестированием.
Постусловие	Прописывают, если систему нужно вернуть в прежний вид после тестирования.
Дополнительные материалы	Помогают проиллюстрировать баг. Например: скриншот или скринкаст.

ЗАГОЛОВОК БАГ-РЕПОРТА

Заголовок баг-репорта исчерпывающе описывает проблему. Это экономит время коллег: они сразу понимают суть, не вникая в детали отчёта. Составляя заголовок,

задавай три вопроса: Что? Где? Когда? Старайся описывать не ожидаемый результат, а фактический.

Пример

ВОПРОС	КАК РАСШИФРОВАТЬ ВОПРОС?	ПРИМЕР
Что?	Что происходит?	Открывается главная страница Яндекса
Где?	В каком месте страницы?	Метро в логотипе
Когда?	При каком действии это происходит?	По клику

ШАГИ ВОСПРОИЗВЕДЕНИЯ

Шаги воспроизведения — это последовательность действий. По ним можно получить ошибку.

В шагах прописывают только ключевые действия.
Например, в задаче про Яндекс.Метро не нужно указывать «Включите компьютер» или «Запустите браузер» — это не относится к продукту. Шаги описывают, начиная с сервиса, — например, сайта или приложения.

Не забывай указывать данные тестирования.
Например, команда тестировщиков проверила поле ввода. Выяснилось, что на символе «&» поле выдаёт ошибку. Другие спецсимволы работают. Написать «Поле ввода ломается на символах» — неправильно. Нужно указать: «Ввести символ &».

Когда закончишь составлять список шагов, пройди по нему как в первый раз. Так ты убедишься, что описание понятное: ссылка — верная, шаги описаны по порядку. Твой отчёт об ошибке может попасть в руки любому: разработчику, менеджеру, заказчику, другому тестировщику, новому человеку из команды. По хорошему отчёту коллеги получат такое же поведение системы.

А ещё указывать ссылку на сервис — хороший тон. Это экономит время коллег: не нужно искать, где ошибка.

ПРИОРИТЕТЫ

Классификация багов

Баги — очень разные.
Их различают **по степени влияния на систему — серьёзности (severity):**

Блокирующий (Blocker), на жаргоне — «блокер». Ни один элемент системы не работает: никто не может ей пользоваться. На сайте интернет-магазина нельзя сделать покупку. Продукт не выполняет свою задачу.

Критический (Critical). Важная часть системы не работает: пользоваться можно, но вероятность сбоя — высокая. В интернет-магазине не работает кнопка «Купить» после авторизации в личном кабинете. Без авторизации всё работает — магазин ещё может принимать заказы.

Высокий (Major). Система работает, но не так, как нужно. Пользоваться можно, элементы активны, но выполняют не то, что задумано. В интернет-магазине по кнопке «Купить» выдают скидку 3% на покупку. Приятно, но это не то, что планировали маркетологи. Кнопка работает неправильно.

Низкий (Minor). С системой всё в порядке, но работать неудобно. Например, дефект в дизайне. В интернет-магазине в разделе «Товары» не работает вертикальная сортировка. Это не мешает пользователям делать заказ — ошибка не такая серьёзная.

Незначительный / **Тривиальный** (Trivial). Баг, который не влияет на работу программы. Например, ошибка в тексте. На сайте интернет-магазина раздел «Чайники» назван «Чайник». Исправить ошибку совсем легко.

Вторая классификация — по скорости исправления, приоритету (priority). Чем выше приоритет, тем быстрее надо исправить дефект. Им часто пользуются, чтобы правильно передать задачи от тестировщиков — разработчикам.

1. **Высокий приоритет (High)** Срочно чинить! Без правок ПО работать не может. Пример: на сайте интернет-магазина нельзя сделать покупку.
2. **Средний приоритет (Medium)** Ошибка некритичная, но без починки ПО не будет работать, как надо. Пример: в разделе «Товары» не работает вертикальная сортировка.
3. **Низкий приоритет (Low)** Можно исправить, когда разобрались с критическими ошибками. Пример: раздел «Чайники» назван «Чайник».

Как классифицировать баги, решает команда. Иногда методики применяют парно: описывают и серьёзность ошибки, и насколько быстро надо чинить.

ОКРУЖЕНИЕ

Обязательные элементы баг-репорта — **окружение и версия приложения.**

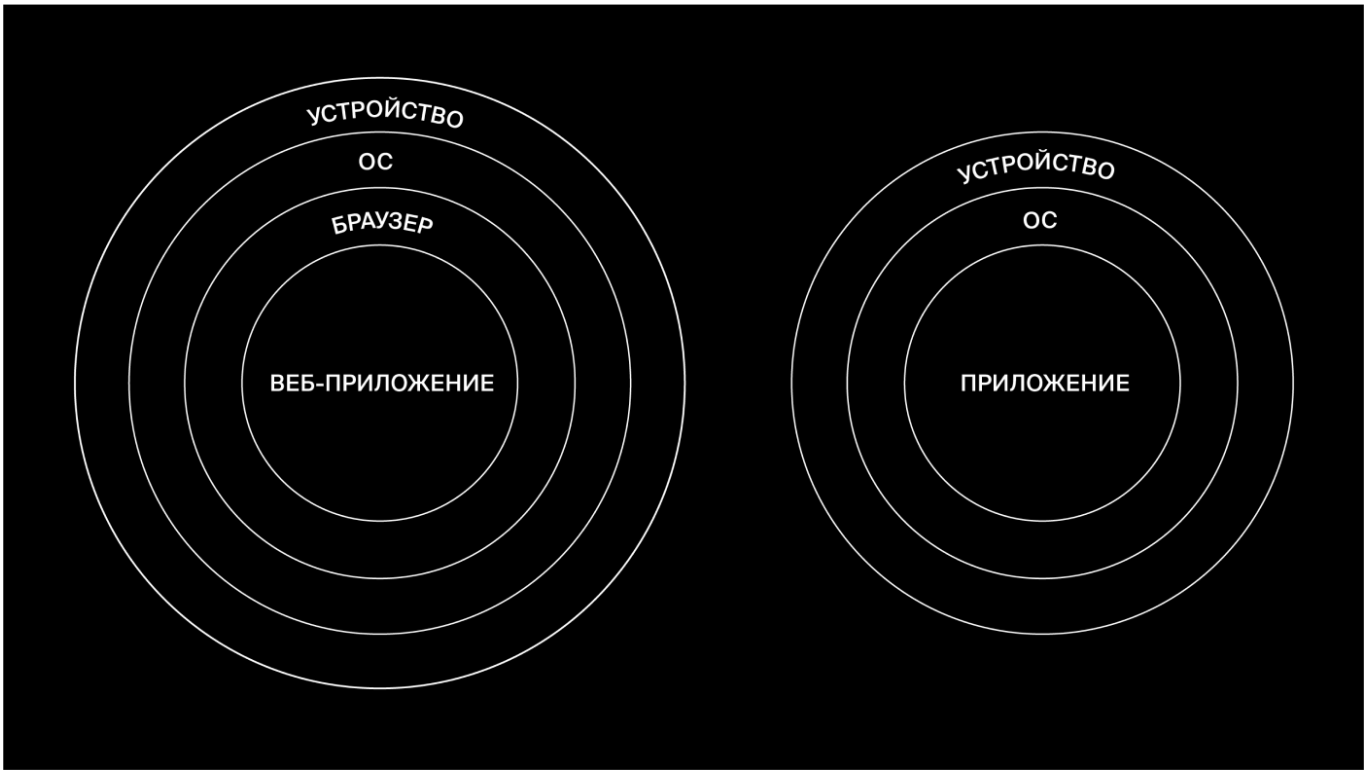
Окружение — это среда, в которой произошла ошибка: информация об операционной системе, браузере и типе устройства. А версия приложения — «номер партии» продукта.

Приложения условно делят на два типа: веб-приложения и нативные. Нативные нужно скачивать и устанавливать. Веб-приложения просто открывают в браузере.

Можно смотреть фильм онлайн: загружаешь страницу в браузере, в которую встроен видеоплеер. Но сервис предлагает и другой путь: установить нативное приложение, чтобы смотреть видео.

Указывать окружение важно: ошибки могут появляться не везде — например, только на iOS или только в Google Chrome на планшете.

Посмотри, что окружает веб- и нативные приложения:



Когда описываешь окружение, иди последовательно. Порядок такой:

1. **Версия приложения** — укажи версию продукта.

2. **Браузер** — укажи версию и название браузера, в котором тестировалось веб-приложение.
3. **ОС** — укажи версию операционной системы: Windows, macOS, Linux.
4. **Устройство** — указывают только для мобильных устройств: планшетов, смартфонов, часов. Не только тип, но и модель. Например, «смартфон Sony».

Если ошибка появляется не во всех случаях:

- Проверь, возникает ли та же ошибка в других окружениях. В баг-репорте можно указать несколько окружений. Например, если ошибка только в Chrome и Firefox, — их и укажи. Если ошибка во всех браузерах, указывай «Воспроизводится везде».
- В отчёте об ошибке нельзя утверждать, что проблема — только в одном окружении. Главное — указать, что баг нашёлся именно там.
- Баг исправили. Финальный шаг — ещё раз убедиться, что он точно пропал.

Какие ОС — популярные?
Самые популярные операционные системы в России можно посмотреть на сервисе Яндекс.Радар:

- Десктопные: https://radar.yandex.ru/desktop
- Мобильные: https://radar.yandex.ru/mobile

ДОПОЛНИТЕЛЬНЫЕ МАТЕРИАЛЫ: ЛОГИ

Дополнительные материалы — ещё один элемент баг-репорта. Они быстрее уточняют и визуализируют ошибку.
Прикрепляют:

- документацию, спецификацию сервиса;
- переписку (пользовательские сообщения о проблеме; детали обсуждения бага с коллегами);
- чек-листы;
- логи;
- скриншоты, скринкасты.

Например, если при загрузке документа на Яндекс.Диск возникла ошибка, приложи файл к баг-репорту. Коллегам не придётся гадать, что именно загружалось.

ЛОГИ

Иногда нужно уточнять:

- Ошибки, которые появились в приложении.
- Какие действия в приложении вызвали баг.

Например, пользователь пытается зайти в приложение под своим логином, но что-то сломалось. Это критическая ситуация: нужно выяснять, что произошло. Пригодятся логи.

Логи — это файлы, в которых хранится информация о системе: список процессов обработки информации, и что в них происходит.
Логи веб-приложения хранятся на сервере в файлах access.log и error.log. В access.log — информация о посещениях страниц, в error.log — об ошибках.

Нативные приложения фиксируют логи по-разному: процесс настраивают разработчики.

Перед тобой баг-репорт, в нём — фрагмент лога. Источник: файл error.log сервера веб-приложения Яндекс.Недвижимость.

[YBUG-6] Не подгружается фоновое изображение формы поиска в Яндекс.Недвижимость

Описание:

На главной странице Яндекс.Недвижимость в форме поиска недвижимости серый фон вместо фонового изображения.

Шаги воспроизведения:

1. Открыть Яндекс.Недвижимость

Фактический результат: Главная страница с формой поиска на сером фоне

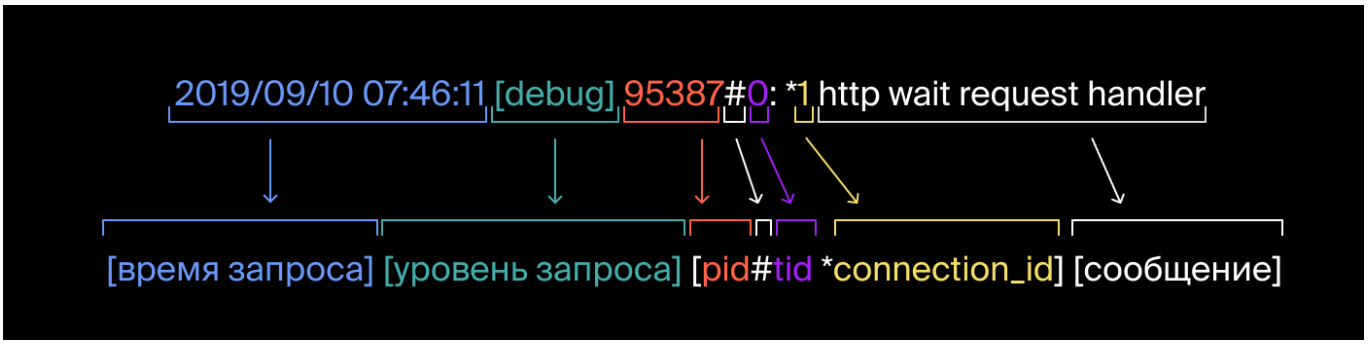
Ожидаемый результат: Главная страница с формой поиска с изображением на фоне

Окружение: браузер Chrome

Приоритет: Критичный

Логи: YBUG-6-error.log

Наборы полей логов зависят от типа сервера и настроек логирования. Как они выглядят:



*pid = process id; tid = thread id — номера. Они помогут отследить, что произошло.
Уровни запросов в логах разные:

- debug — подробные уведомления, которые помогают отладить систему;
- info — сообщения о процессах в системе;
- notice — системные уведомления;
- warn — предупреждение от системы;
- error — баги.

Логи веб-приложения хранятся на сервере: к нему понадобится доступ. Важно прикреплять к баг-репорту документ с логами, а не скриншот — копировать быстрее, чем переписывать с картинки.

Как правильно называть вложения?
Из названия должно быть сразу ясно, что в дополнительном материале и к какому баг-репорту относится этот файл.

Применяй универсальный алгоритм. Так ты подберёшь ёмкие названия вложений.
{УИД бага}-{название сервиса}-{суть проблемы}.png

ОРАКУЛ

Как оценить, насколько ошибка серьёзна? Хорошо, когда есть требования. Но что делать, если их нет или они неточные?

Оракулы. Это инструменты, которые помогают достроить картинку проблемы. Они подходят в отдельных ситуациях и не указывают на проблему точно. Скорее подсказывают, где скрывается баг.

Полезная мнемоника
Оракулы легко запомнить по мнемонике **FEW HICCUPS**:

Иллюстрация мнемоники поможет разобраться, где искать ошибку.

Блок 1. Технические оракулы

- **История.** Сравни новую версию продукта с предыдущей.
- **Заявления.** Проверь, что система работает так, как о ней говорят:
  - документация (требования, результаты проверки, регламенты);
  - твоя команда (выводы с встреч, планёрок, долгосрочного планирования).
- **Объяснимость.** Убедись в том, что любой шаг работы системы можно объяснить.
- **Известность.** Проверь, что в продукте нет ошибок, которые уже нашли раньше.

Блок 2. Продуктовые оракулы

- **Имидж**. Убедись, что в продукте есть ценности, которые ассоциируются с брендом.

Например, компания занимается экологическим активизмом, но продаёт сувениры из неперерабатываемого пластика.

- **Конкуренты**. Проверь, что у конкурентов продукт выстроен по той же системе.

Например, сайт доставки цветов. Заказать цветы можно только на свой адрес при предъявлении паспортных данных. Вряд ли это будет востребовано, когда конкуренты рассылают букеты куда угодно по желанию клиента.

- **Продукт**. Проверь, что все элементы системы согласуются между собой и органичны.

Например, в Яндекс.Метро кнопка «+» добавляет в приложение корзину и предлагает купить станцию. Это элемент, который не имеет отношения к логике сервиса — Яндекс.Метро помогает оптимизировать маршруты.

- **Ожидания пользователя**. Предугадай ожидания пользователя и учти их.

Например, сайт, который позволяет заказать пиццу. Но заказ нельзя оплатить картой — это неудобно клиенту.

Блок 3. Логические оракулы

- **Цель**. Проверь, что продукт работает по назначению и выполняет все шаги.

Например, в интернет-магазине мебели можно автоматически посчитать стоимость заказа, а сделать сам заказ нельзя.

- **Стандарты и законы**. Проверь, что продукт не нарушает никакие требования и нормы закона.

Например, в интернет-магазине в России продаётся алкоголь с курьерской доставкой. Согласно последним поправкам, это запрещено.

- **Мир**. Проверь, что продукт работает по законам логики.

Например, нельзя купить отрицательное количество товаров.

ЗАДАЧА: пройти ТЕСТ