Н.В. Выговская, ст.пр. Е.В. Макаров, студ.
(Белорусско-Российский университет, г. Могилев)

ТЕХНОЛОГИЯ ОБНАРУЖЕНИЯ ОШИБОК ВО ФРОНТЕНД-ПРИЛОЖЕНИЯХ НА JAVASCRIPT

Устранение ошибок, возникших у пользователя, во фронтенд-приложениях на JavaScript может оказаться непростой задачей, так как они возникают в браузере и на устройстве пользователя, к чему, зачастую, у разработчиков нет доступа.

Допустим, мы предприняли максимально возможное количество необходимых мер по недопущению ошибок во время разработки и сборки конечной версии проекта. Тем не менее, ошибки все равно могут проникать в код, ушедший в релиз. Вследствие, требуется как-то узнавать о наличии таковых и принимать молниеносные меры по исправлению. Просить пользователей приложения открывать консоль браузера и делать скриншоты — худший из возможных вариантов. Поэтому к проекту неплохо, а зачастую и необходимо, подключить логирование ошибок.

Смысл любой технологии, предназначенной для логирования, прост: на каждое событие window.onerror или аналогичное ему для технологий, не взаимодействующих с браузером пользователя, или же в каждый переход исполнения кода в блок catch выполняется простой AJAX-запрос на специально выделенный адрес сервера, в тело которого кладется информация об ошибке. Далее потребуется инструмент, который быстро оповестит техподдержку и разработчиков о наличии новых ошибок и позволит эффективно работать с ними. Самый популярный из таких инструментов для фронтенда — Sentry.

Система логирования Sentry позволяет собирать, группировать и представлять ошибки в реальном времени. Есть сборки для разных языков, в том числе и для JavaScript [1]. Проект предоставляет платный доступ с расширенными возможностями для бизнеса, однако можно попробовать его основные возможности на бесплатном тестовом аккаунте.

Подключать Sentry можно как непосредственно в HTML-файле, так и в компонентах, выполненных на одном из популярных фреймворков: React, Vue, Angular, Ember и других.

В Sentry присутствует ниже перечисленный функционал:

- обновление списка ошибок в режиме реального времени;
- группировка и сортировка полученных ошибок, например по частоте появления;
- фильтрация ошибок по статусу, уровню логирования, источнику и другим параметрам;
- возможность реинкарнации ошибки. Если ошибка была помечена как решенная и появилась снова, то она снова вносится в список и учитывается в отдельном потоке;
- отправка e-mail, sms или чат-сообщений, в случае получения новой ошибки;

- возможность запроса Feedback’а пользователя;
- возможность интегрирации с такими системами как JIRA, GitHub, Bitbucket и другими.

Помимо вышеназванного Sentry умеет рисовать графики событий по возникшим "исключениям". Для конкретного "исключения" умеет выводить заголовки, cookies и прочее, стек с контекстом (а не голый как при получении на почту), окружение пользователя (какой был браузер, ОС и т.п.), данные о авторизованном пользователе (для запросов), информацию о установленных приложениях и т.п.

Таким образом, Sentry является незаменимым инструментом. На практике часто приходится сталкиваться со случаями, когда ваше приложение уже ушло в релиз, а пользователи сталкиваются с непредвиденными ошибками.

В случае, если Sentry не настроен – придется вывести приложение из релиза и потратить огромное количество времени на воспроизведение ошибки. Это не критично для небольших приложений без монетизации. Для сложных же систем подобное явление может стать критическим. Можно не только потерять деньги, но и лишиться огромной пользовательской базы и доверия со стороны клиентов.

Однако, если Sentry настроен верно, мы сможем увидеть не только ошибку, но и точное место, где она появилась. Это позволит сделать нам хотфикс в ближайшие минуты или часы, спасти наше приложение и не потерять клиентов.

Так же, с помощью Sentry можно обнаруживать ошибки, появившиеся вследствие мердж-конфликтов после размещения приложения в интернете. Если мы видим ошибку, которая появилась вновь, после того как вышел хотфикс нам стоит проверить, верно ли прошла сборка нашего приложения или же, в случае мобильного приложения, выслать пользователю нотификацию с просьбой обновиться на актуальную версию.

Ярким примером подобного является один из случаев, с которым пришлось столкнуться на реальном проекте. После выпуска очередного планового обновления приложения Sentry отобразил ошибку, продемонстрированную на рисунке 1.

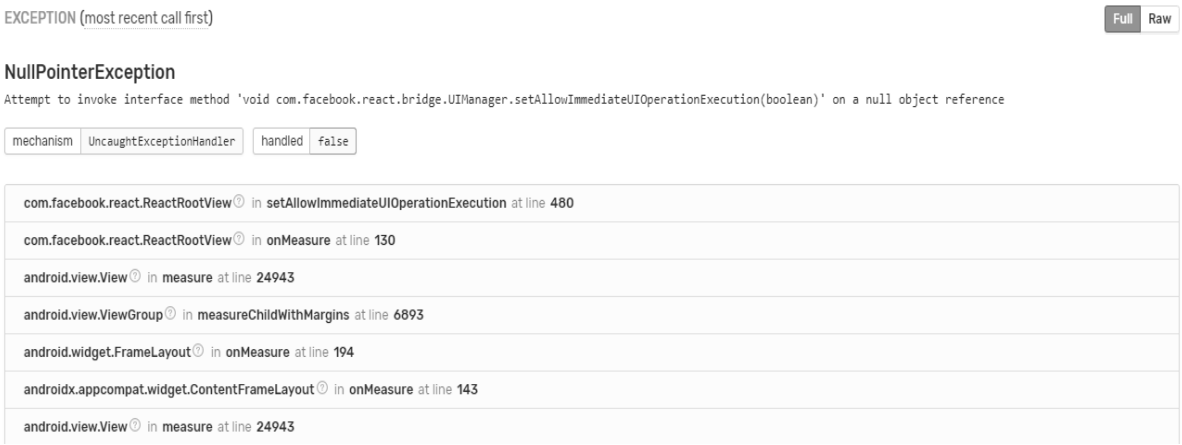

Рисунок 1 – Нативная ошибка приложения.

Как удалось выяснить позже, используя данные из Sentry, отображенные на рисунке 2, для некоторых новых андроид устройств не была заблокирована ориентация экрана на портретном режиме отображения, что, в свою очередь, вызывало фатальную ошибку приложения [2] из-за невозможности адаптировать некоторые нативные [3] элементы приложения под быстро меняющийся размер экрана. Благодаря Sentry нам удалось быстро выявить эту ошибку и уже в ближайший час выпустить исправленную версию до того, как обновилось большое число пользователей.

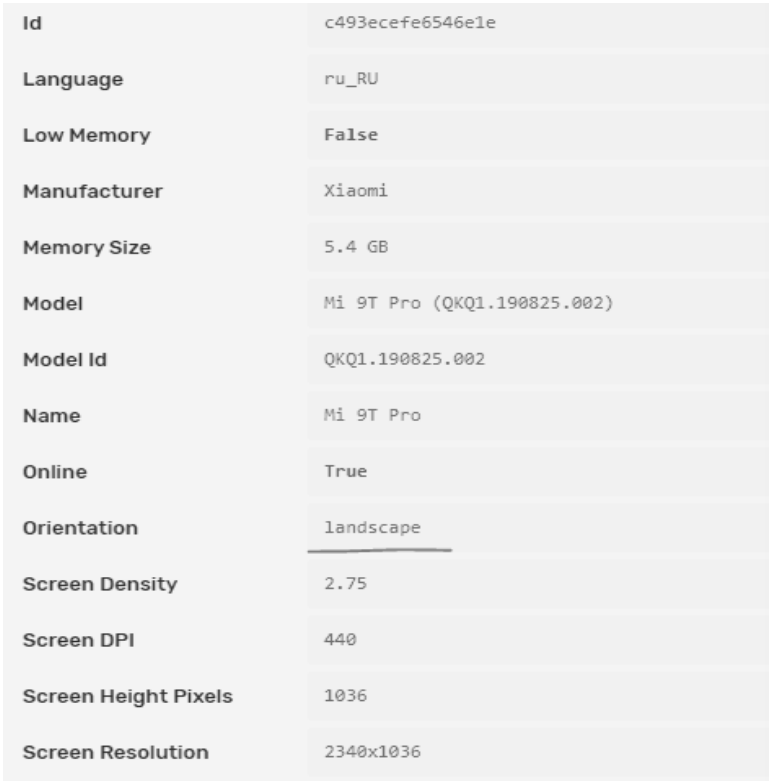

Рисунок 2 – Отображение данных пользователя, столкнувшегося с ошибкой.

Для данной проблемы принято считать, что так называемой «серебряной пули» или же панацеи не существует. Однако, в случае Sentry – это именно тот случай, когда мы имеем технологию, значительно упрощающую жизнь и не имеющую недостатков при верном использовании. Технология опробована на трёх реальных проектах и подтвердила высокую эффективность.

Литература

1. Климов Александр JavaScript на примерах; БХВ-Петербург - М., 2017. - 812 с.

2. Application Monitoring and Error Tracking Software | Sentry. Официальный сайт и документация инструмента Sentry w3. – https://www.sentry.io/.

3. React Native · A framework for building native apps using React. Официальный сайт и документация фреймворка React-Native w3. – https://reactnative.dev/.