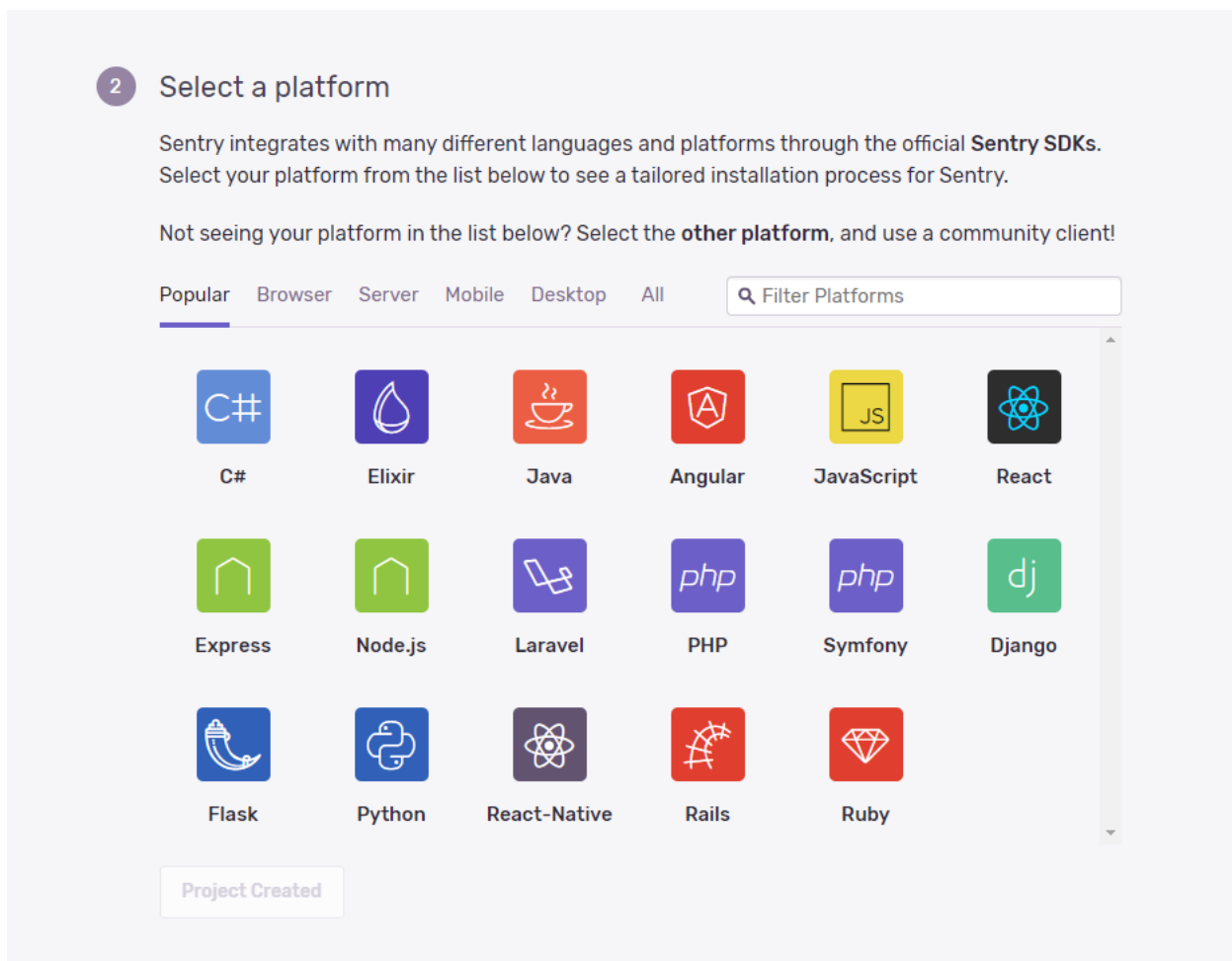# Sentry在vue项目中的使用

应用场景

Sentry是提供基于云端的错误监控报警系统，只要在项目代码中集成Sentry提供的sdk，便可在云端控制台查看，分析，处理错误，当然也可以部署在自有的服务器上，自己实现运维管理。

Sentry适用于多种语言，也为各种前后端热门的框架提供了配套的SDK。

注册创建应用

首页我们注册账号，新账户会有一个14天的免费使用期，过了试用期后可选择付费套餐或者使用功能受限的个人免费版。选择相应的框架，选择后会生成一个简单的安装指南。我们在"Browser"下找到了"vue"，然后开始安装。

安装sdk

```
1  # Using yarn
```

```
2  $ yarn add @sentry/browser
3
4  # Using npm
5  $ npm install @sentry/browser
```

```
1  # Using yarn
2  yarn add @sentry/integrations
3
4  # Using npm
5  npm install @sentry/integrations
```

通过安装 *sentry/browser*，可以捕捉到浏览器环境js的错误，而安装 *sentry/intergrations*，通过vue全局的*config.errorHandler*可以定位错误是从哪个vue组件中的具体哪个属性引起。

最后在main.js中初始化监控

```
1  import Vue from 'vue'
2  import * as Sentry from '@sentry/browser';
3  import { Vue as VueIntegration } from '@sentry/integrations';
4
5  Sentry.init({
6    dsn: 'https://******************@o409167.ingest.sentry.io/5281135',
7    integrations: [new VueIntegration({Vue, attachProps: true, logErrors: true})],
8    debug:true,
9    environment: 'dev'
10  });
```
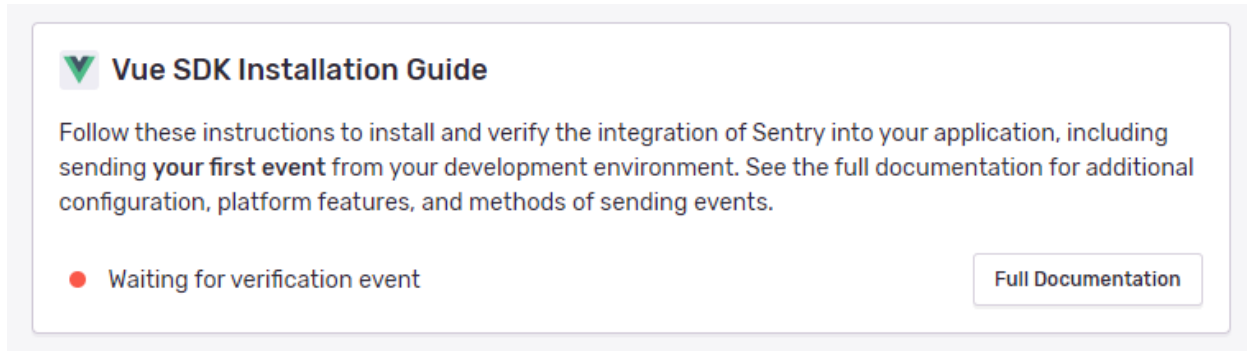
init方法中的配置参数主要有：

- dsn: 相当于密钥，也是区别项目的唯一id
- debug: 是否记录debug信息
- release: 记录当前的项目发布版本，便于多版本使用中的问题定位
- environment: 预设环境，便于在控制台过滤
- ...

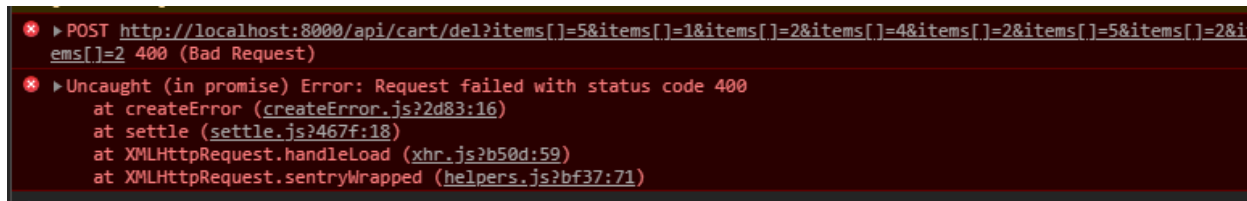更具体的配置可参考文档：https://docs.sentry.io/error-reporting/configuration/?platform=browser

上述是浏览器环境的通用配置，而在vue项目中integrations的配置，记得添加 logErrors: true 的配置，不然在开发环境中，无法在console中看见错误信息。
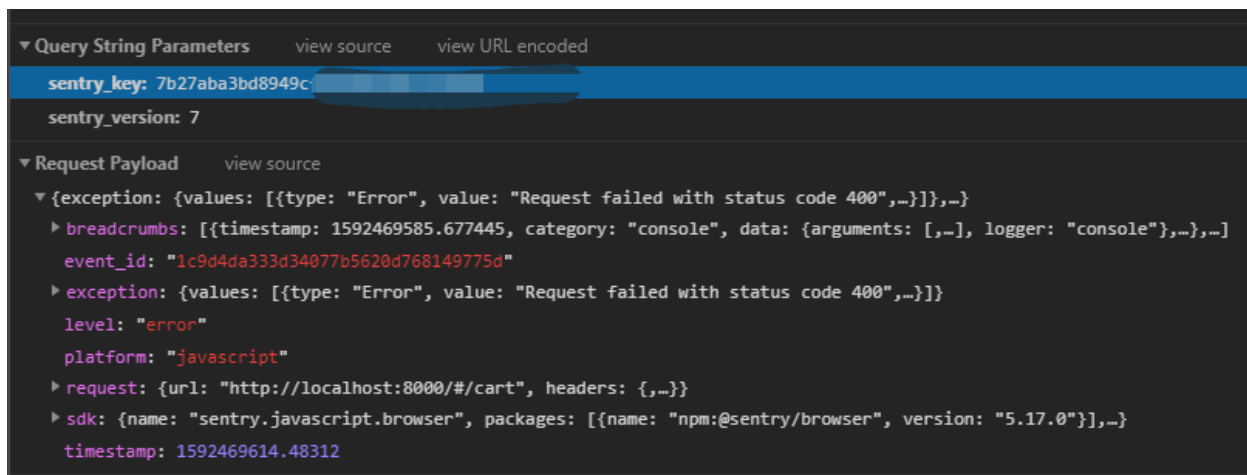
验证结果

可以看到，网页一直在等待着错误信息，验证我们是否安装成功

我们在开发环境中，故意产生一个错误

我们post一个接口后发生了400错误，sentry立刻帮助我们捕捉并上报，可以看下具体上报的参数。

- breadcrumbs 包含了用户的操作步骤，能让我们更快地复现bug。
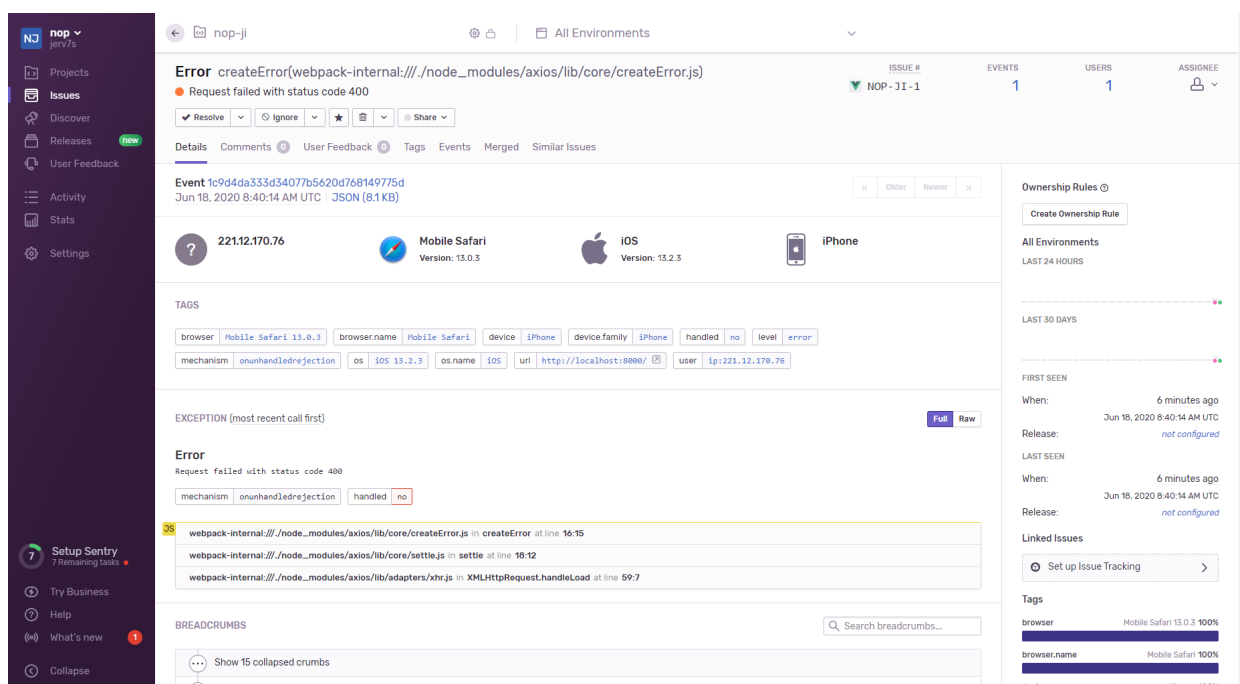- exception 包含错误的具体信息
- request 是报错接口的详情
- …

然后我们在页面上看到了服务端更新了页面状态，上报事件已被接收。我们点击进入控制台看下：

控制台上的信息十分详细，除了上报请求内的参数外，还把http头的ua等相关信息解析出来了，也可对比不同机型发现兼容性问题。

我们在代码中修改一些，看看vue组件的错误如何被捕捉。

See more tips at https://vuejs.org/guide/deployment.html

⊗ ▶[Vue warn]: Error in created hook: "TypeError: Cannot read property 'id' of undefined"

  found in

---> <Info> at src/page/user/order/info.vue
       <App> at src/App.vue
         <Root>

⊗ ▶TypeError: Cannot read property 'id' of undefined
    at VueComponent.created (info.vue?4445:81)
    at invokeWithErrorHandling (vue.runtime.esm.js?2b0e:1854)
    at callHook (vue.runtime.esm.js?2b0e:4219)
    at VueComponent.Vue._init (vue.runtime.esm.js?2b0e:5008)
    at new VueComponent (vue.runtime.esm.js?2b0e:5154)
    at createComponentInstanceForVnode (vue.runtime.esm.js?2b0e:3283)
    at init (vue.runtime.esm.js?2b0e:3114)
    at merged (vue.runtime.esm.js?2b0e:3301)
    at createComponent (vue.runtime.esm.js?2b0e:5978)
    at createElm (vue.runtime.esm.js?2b0e:5925)
    at updateChildren (vue.runtime.esm.js?2b0e:6216)
    at patchVnode (vue.runtime.esm.js?2b0e:6319)
    at VueComponent.patch [as __patch__] (vue.runtime.esm.js?2b0e:6482)
    at VueComponent.Vue._update (vue.runtime.esm.js?2b0e:3948)
    at VueComponent.updateComponent (vue.runtime.esm.js?2b0e:4066)
    at Watcher.get (vue.runtime.esm.js?2b0e:4479)
    at Watcher.run (vue.runtime.esm.js?2b0e:4554)
    at flushSchedulerQueue (vue.runtime.esm.js?2b0e:4310)
    at Array.eval (vue.runtime.esm.js?2b0e:1980)
    at flushCallbacks (vue.runtime.esm.js?2b0e:1906)

▼ Request Payload   view source
▼ {,…}
  ▶ breadcrumbs: [{timestamp: 1592470496.865464, category: "console", data: {arguments: [,…], logger: "console"},…},…]
  ▼ contexts: {vue: {componentName: "Info", propsData: "[undefined]", lifecycleHook: "created hook"}}
    ▼ vue: {componentName: "Info", propsData: "[undefined]", lifecycleHook: "created hook"}
        componentName: "Info"
        lifecycleHook: "created hook"
        propsData: "[undefined]"
    event_id: "c1d49c556d494c1aa3eb4d4d6fb4150e"
  ▼ exception: {values: [{type: "TypeError", value: "Cannot read property 'id' of undefined", stacktrace: {,…},…}]}
    ▼ values: [{type: "TypeError", value: "Cannot read property 'id' of undefined", stacktrace: {,…},…}]
      ▼ 0: {type: "TypeError", value: "Cannot read property 'id' of undefined", stacktrace: {,…},…}
        ▶ mechanism: {handled: true, type: "generic"}
        ▶ stacktrace: {,…}
          type: "TypeError"
          value: "Cannot read property 'id' of undefined"
    level: "error"
    platform: "javascript"
  ▶ request: {url: "http://localhost:8000/#/user/order/info/2", headers: {,…}}
  ▶ sdk: {name: "sentry.javascript.browser", packages: [{name: "npm:@sentry/browser", version: "5.17.0"}],…}
    timestamp: 1592470496.941649

⚠ console   [Vue warn]: Error in created hook: "TypeError: Cannot read property 'id' of undefined"   08:54:56

               found in

               ---> <Info> at src/page/user/order/info.vue
                      <App> at src/App.vue
                        <Root>

    logger   console
    arguments ["[Vue warn]: Error in created hook: \"TypeError: Cannot read property 'id' of undefined\"\n\nfound in\n\n---> <Info> at src/page/user/order/i
               nfo.vue\n   <App> at src/App.vue\n   <Root>"]

⚠ console   TypeError: Cannot read property 'id' of undefined   08:54:56

    arguments [{"message":"Cannot read property 'id' of undefined","name":"TypeError","stack":"TypeError: Cannot read property 'id' of undefined\n   at Vue
               Component.created (webpack-internal:///./node_modules/cache-loader/dist/cjs.js?!./node_modules/babel-loader/lib/index.js!./node_modules/cache-
               loader/dist/cjs.js?!./node_modules/vue-loader/lib/index.js?!./src/page/user/order/info.vue?vue&type=script&lang=js&:75:41)\n   at invokeWithE
               rrorHandling (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:1853:57)\n   at callHook (webpack-internal:///./node_modules/vu
               e/dist/vue.runtime.esm.js:4213:7)\n   at VueComponent.Vue._init (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:4998:5)\n
               at new VueComponent (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:5144:12)\n   at createComponentInstanceForVnode (webpack-
               internal:///./node_modules/vue/dist/vue.runtime.esm.js:3280:10)\n   at init (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:3
               111:45)\n   at merged (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:3298:5)\n   at createComponent (webpack-internal:///./
               node_modules/vue/dist/vue.runtime.esm.js:5968:9)\n   at createElm (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:5915:9)\n
               at updateChildren (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:6309:29)\n   at VueComponent.patch [as __patch__] (webpack-internal:///./node_modules/vue/dist/vue.runtime.es
               m.js:6472:9)\n   at VueComponent.Vue._update (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:3942:19)\n   at VueComponent.up
               dateComponent (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:4060:10)\n   at Watcher.get (webpack-internal:///./node_module
               s/vue/dist/vue.runtime.esm.js:4473:25)\n   at Watcher.run (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:4548:22)\n   at fl
               ushSchedulerQueue (webpack-internal:///./node_modules/vue/dist/vue.runtime.esm.js:4304:13)\n   at Array..."}]

这是一个常见的错误,可以看到contexts里可以识别到发生错误的组件的名称，错误产生的事件，和错误的属性是哪一个。控制台的面包屑也是完完全全复原了控制台中的错误堆栈。

生产环境问题定位

刚才的尝试是在开发环境中的，所以错误堆栈产生的原因都能清晰地定位到具体代码行，如果实在生产环境中，我们需要上传sourcemap。

安装命令行工具

可以在github上下载相应系统的工具，并且重命名为 sentry-cli

https://github.com/getsentry/sentry-cli/releases/

生成配置文件

在工具目录下执行

```
1  sentry-cli login
```

要求我们填入token，我们可以在网页上的控制台找到，然后命令行工具会生成一个 .sentryclirc 的配置文件，放在项目根目录即可。配置文件中还需要填org和project，与注册账号时填写的信息一致。

当然你也可以手动创建，就像这样

```
1  [auth]
2  token=************
3  [defaults]
4  project=****
5  org=****
```

上传插件配置

Sentry提供一个webpack插件，能使我们在工程打包时把sourcemap文件上传。

```
1  $ npm install --save-dev @sentry/webpack-plugin
2  //or
3  $ yarn add --dev @sentry/webpack-plugin
```

我们在打包时，发现报错，找不到对应的map文件。

这里需要将map文件和js文件输出在同一个目录，所以在vue.config.js中，我们需要修改如下：

productionSourceMap: true

这样map文件和js文件生成在同一个目录，sentry的插件会自动识别并上传。注：不要用SourceMapDevToolPlugin这个插件，生成的文件路径在根目录，插件识别不了。

献上插件配置

```
1        new SentryWebpackPlugin({
2          release: "202006191603",
3          include: './dist/',
4          urlPrefix: "http://127.0.0.1:8080",
5          ignoreFile: '.sentrycliignore',
6          ignore: ['node_modules', 'vue.config.js'],
7          configFile: 'sentry.properties'
8        })
```

注意的是：include 和 urlPrefix ，使得最后线上的地址能够访问得到js文件

```
 Building for production...
Source Map Upload Report
  Minified Scripts
    http://127.0.0.1:8080/js/app.js (sourcemap at app.js.map)
    http://127.0.0.1:8080/js/chunk-03806d90.js (sourcemap at chunk-03806d90.js.map)
    http://127.0.0.1:8080/js/chunk-060b1ef6.js (sourcemap at chunk-060b1ef6.js.map)
    http://127.0.0.1:8080/js/chunk-184f4b3e.js (sourcemap at chunk-184f4b3e.js.map)
    http://127.0.0.1:8080/js/chunk-2d0c1175.js (sourcemap at chunk-2d0c1175.js.map)
    http://127.0.0.1:8080/js/chunk-2d0c4a05.js (sourcemap at chunk-2d0c4a05.js.map)
    http://127.0.0.1:8080/js/chunk-2d0d374d.js (sourcemap at chunk-2d0d374d.js.map)
    http://127.0.0.1:8080/js/chunk-2d0e4add.js (sourcemap at chunk-2d0e4add.js.map)
    http://127.0.0.1:8080/js/chunk-2d0e6363.js (sourcemap at chunk-2d0e6363.js.map)
    http://127.0.0.1:8080/js/chunk-2d210098.js (sourcemap at chunk-2d210098.js.map)
    http://127.0.0.1:8080/js/chunk-2d230890.js (sourcemap at chunk-2d230890.js.map)
    http://127.0.0.1:8080/js/chunk-5105bcd9.js (sourcemap at chunk-5105bcd9.js.map)
    http://127.0.0.1:8080/js/chunk-56ecaf18.js (sourcemap at chunk-56ecaf18.js.map)
    http://127.0.0.1:8080/js/chunk-5ae84624.js (sourcemap at chunk-5ae84624.js.map)
    http://127.0.0.1:8080/js/chunk-6ba1d848.js (sourcemap at chunk-6ba1d848.js.map)
    http://127.0.0.1:8080/js/chunk-700aff4a.js (sourcemap at chunk-700aff4a.js.map)
    http://127.0.0.1:8080/js/chunk-7772be04.js (sourcemap at chunk-7772be04.js.map)
    http://127.0.0.1:8080/js/chunk-7821aaf4.js (sourcemap at chunk-7821aaf4.js.map)
    http://127.0.0.1:8080/js/chunk-7ad9a0a8.js (sourcemap at chunk-7ad9a0a8.js.map)
    http://127.0.0.1:8080/js/chunk-7f33b350.js (sourcemap at chunk-7f33b350.js.map)
    http://127.0.0.1:8080/js/chunk-85b30404.js (sourcemap at chunk-85b30404.js.map)
    http://127.0.0.1:8080/js/chunk-983f20a4.js (sourcemap at chunk-983f20a4.js.map)
    http://127.0.0.1:8080/js/chunk-99db9ff0.js (sourcemap at chunk-99db9ff0.js.map)
    http://127.0.0.1:8080/js/chunk-b018d584.js (sourcemap at chunk-b018d584.js.map)
    http://127.0.0.1:8080/js/chunk-b0b432e8.js (sourcemap at chunk-b0b432e8.js.map)
    http://127.0.0.1:8080/js/chunk-f6e19b6e.js (sourcemap at chunk-f6e19b6e.js.map)
    http://127.0.0.1:8080/js/chunk-vendors.js (sourcemap at chunk-vendors.js.map)
  Source Maps
    http://127.0.0.1:8080/js/app.js.map
    http://127.0.0.1:8080/js/chunk-03806d90.js.map
    http://127.0.0.1:8080/js/chunk-060b1ef6.js.map
    http://127.0.0.1:8080/js/chunk-184f4b3e.js.map
    http://127.0.0.1:8080/js/chunk-2d0c1175.js.map
    http://127.0.0.1:8080/js/chunk-2d0c4a05.js.map
    http://127.0.0.1:8080/js/chunk-2d0d374d.js.map
    http://127.0.0.1:8080/js/chunk-2d0e4add.js.map
    http://127.0.0.1:8080/js/chunk-2d0e6363.js.map
    http://127.0.0.1:8080/js/chunk-2d210098.js.map
    http://127.0.0.1:8080/js/chunk-2d230890.js.map
    http://127.0.0.1:8080/js/chunk-5105bcd9.js.map
    http://127.0.0.1:8080/js/chunk-56ecaf18.js.map
    http://127.0.0.1:8080/js/chunk-5ae84624.js.map
    http://127.0.0.1:8080/js/chunk-6ba1d848.js.map
    http://127.0.0.1:8080/js/chunk-700aff4a.js.map
    http://127.0.0.1:8080/js/chunk-7772be04.js.map
    http://127.0.0.1:8080/js/chunk-7821aaf4.js.map
    http://127.0.0.1:8080/js/chunk-7ad9a0a8.js.map
    http://127.0.0.1:8080/js/chunk-7f33b350.js.map
    http://127.0.0.1:8080/js/chunk-85b30404.js.map
    http://127.0.0.1:8080/js/chunk-983f20a4.js.map
    http://127.0.0.1:8080/js/chunk-99db9ff0.js.map
    http://127.0.0.1:8080/js/chunk-b018d584.js.map
    http://127.0.0.1:8080/js/chunk-b0b432e8.js.map
    http://127.0.0.1:8080/js/chunk-f6e19b6e.js.map
    http://127.0.0.1:8080/js/chunk-vendors.js.map
 Building for production...
```

最后，控制台中就可以看到源码解析的文件了。

```
     TypeError
     Cannot read property 'id' of undefined

     mechanism   generic      handled   yes

 JS    src/page/product/detail.vue in call at line 148:39

      143.         this.$router.push("/cart");
      144.       });
      145.     }
      146.   },
      147.   created() {
      148.     this.itemId = this.$router.params.id;
      149.     getDetail(this.itemId).then(data => {
      150.       this.goods = data;
      151.     });
      152.   }
      153. };
```

其他功能介绍

- 配置邮件提醒，让你第一时间收到通知进行处理
- 根据不同条件筛选错误，默认是未解决的错误，聚焦未解决的错误
- 可以添加成员，并根据错误信息，设置分配规则，分配错误给owner
- 可以忽略错误，或者设置忽略的阈值（包括忽略时长，发生次数和影响人数）
- 分享错误详情，请项目外高手看一看错误信息，帮助解决

自托管搭建

由于免费版的功能限制，比如只能一个人玩，一个月的错误数量也是限制在5000以下等，我们考虑可以自己来托管Sentry的服务端。

在docker环境中运行监控服务端

可按照官方github仓库部署到私有云：<u>https://github.com/getsentry/onpremise</u>

- 环境要求：Docker 17.05.0+ / Compose 1.23.0+ / RAM 2400MB+

克隆git仓库，并根据需要配置相关文件，也可直接执行安装脚本。

config.yml

可以配置邮件设置（错误上报时邮件提醒），系统设置（重置密钥，redis集群设置），文件存储设置（修改文件保存位置如文件系统或s3云存储），github,slack的插件集成

sentry.conf.py

django框架的相关配置，诸如数据库链接，缓存链接，消息队列，web服务器等其他配置

.env

环境相关的配置项

配置完后相关文件之后，就可以执行安装了，安装完毕后，会在9000端口提供控制台面板web服务，进入后即可开始项目的配置与安装。

it_sentry地址： http://itsentry.yunjiglobal.com/