

uniapp 安卓腾讯bugly原生插件

目 录

使用说明	1
使用方法	1.1
更新日志	1.2
插件方法	2
初始化	2.1
设置设备id	2.2
设置设备型号	2.3
设置App版本	2.4
设置渠道	2.5
设置包名	2.6
设置标签	2.7
设置用户ID	2.8
模拟Native崩溃	2.9

使用方法

介绍

腾讯Bugly SDK可以帮助你发现多类异常问题，如崩溃，ANR，OOM，内存使用异常，应用操作卡顿，启动耗时过长等质量问题

联系作者

关注微信公众号可联系作者



插件地址

<https://ext.dcloud.net.cn/plugin?id=17813>

用法

在需要使用插件的页面加载以下代码

```
const module = uni.requireNativePlugin("leven-qqBugly-BuglyModule");
```

页面内容

```
<template>
  <view>
    <uni-card title="安卓腾讯bugly性能监控原生插件">
```

```

<view style="margin-bottom: 20px;">
  <button type="primary" @click="init">初始化</button>
  <button type="primary" @click="setDeviceID">设置设备id</button>
  <button type="primary" @click="setDeviceModel">设置设备型号</button>
  <button type="primary" @click="setAppVersion">设置App版本</button>
  <button type="primary" @click="setAppChannel">设置渠道</button>
  <button type="primary" @click="setAppPackage">设置包名</button>
  <button type="primary" @click="setUserSceneTag">设置标签</button>
  <button type="primary" @click="setUserId">设置用户ID</button>
  <button type="primary" @click="testNativeCrash">模拟Native崩溃</button>
</view>
</uni-card>
<uni-card class="uni-card-box" title="日志">
  <view><text style="font-size: 14px; flex-wrap: wrap;">{{logStr}}</text></view>
</uni-card>
</view>
</template>

<script>
  const module = uni.requireNativePlugin("leven-qqBugly-BuglyModule");
  export default {
    data() {
      return {
        logStr: "",
      },
    },
    methods: {
      // 初始化
      init() {
        module.init({
          // 【必需】在Bugly 注册产品的appID
          appID: "09bfd7592d",
          // 【非必须】设备id
          deviceId: "123",
          // 【非必须】设备型号
          deviceModel: "huawei",
          // 【非必须】渠道
          appChannel: "android",
          // 【非必须】app版本号
          appVersion: "v1.0.0",
          // 【非必须】app包名
          appPackage: "com.uni.tx.bugly.test",
          // 【非必须】初始化延迟，默认：10000
          appReportDelay: 20000,
          // 【非必须】标签，必须为数字
          tag: 111,
          // 【非必须】设置anr时是否获取系统trace文件，默认：false
          enableCatchAnrTrace: false,
          // 【非必须】设置是否获取anr过程中的主线程堆栈，默认为true
          enableRecordAnrMainStack: true,
        });
      },
    },
  };

```

```
        // 【非必须】用户id
        userId: ""
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置设备id
    setDeviceID() {
      module.setDeviceID({
        deviceId: "123"
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置设备型号
    setDeviceModel() {
      module.setDeviceModel({
        deviceModel: "123"
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置App版本
    setAppVersion() {
      module.setAppVersion({
        appVersion: "123"
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置渠道
    setAppChannel() {
      module.setAppChannel({
        appChannel: "123"
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置包名
    setAppPackage() {
      module.setAppPackage({
        appPackage: "123"
      }, res => {
        this.writeLog(JSON.stringify(res))
      })
    },
    // 设置标签
    setUserSceneTag() {
      module.setUserSceneTag({
        tag: 123
      })
    }
  }
}
```

```
    }, res => {
      this.writeLog(JSON.stringify(res))
    })
  },
  // 设置用户ID
  setUserId() {
    module.setUserId({
      userId: "123"
    }, res => {
      this.writeLog(JSON.stringify(res))
    })
  },
  // 模拟Native崩溃
  testNativeCrash() {
    module.testNativeCrash(res => {
      this.writeLog(JSON.stringify(res))
    })
  },
  // 写日志
  writeLog(str) {
    console.log(str);
    let logStr = uni.$lv.date.format(null, "yyyy-mm-dd hh:MM:ss") + " " + str
    this.logStr = logStr + this.logStr;
  }
}
}
</script>

<style>

</style>
```

更新日志

2024-04-25

首次发布

初始化

方法名

init

用法

- 用法如下：

```
module.init({
  // 【必需】在Bugly 注册产品的appID
  appID: "09bfd7592d",
  // 【非必须】设备id
  deviceId: "123",
  // 【非必须】设备型号
  deviceModel: "huawei",
  // 【非必须】渠道
  appChannel: "android",
  // 【非必须】app版本号
  appVersion: "v1.0.0",
  // 【非必须】app包名
  appPackage: "com.uni.tx.bugly.test",
  // 【非必须】初始化延迟，默认：10000
  appReportDelay: 20000,
  // 【非必须】标签，必须为数字
  tag: 111,
  // 【非必须】设置anr时是否获取系统trace文件，默认：false
  enableCatchAnrTrace: false,
  // 【非必须】设置是否获取anr过程中的主线程堆栈，默认为true
  enableRecordAnrMainStack: true,
  // 【非必须】用户id
  userId: ""
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
appID	String	是	无	腾讯Bugly申请的appID
deviceId	Integer	否	无	设备id

参数名	参数类型	是否必填	默认值	参数描述
deviceModel	String	否	无	设备型号
appChannel	String	否	无	渠道
appVersion	String	否	无	app版本号
appPackage	String	否	无	包名
appReportDelay	Integer	10000	无	初始化延迟，单位：毫秒
tag	Integer	否	无	标签
enableCatchAnrTrace	Boolean	否	false	设置anr时是否获取系统trace文件
enableRecordAnrMainStack	Boolean	否	true	设置是否获取anr过程中的主线程堆栈
userId	String	否	无	用户id

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置设备id

方法名

setDeviceID

用法

- 用法如下：

```
module.setDeviceID({
  deviceId: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
deviceId	Integer	否	无	设备id

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置设备型号

方法名

setDeviceModel

用法

- 用法如下：

```
module.setDeviceModel({
  deviceModel: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
deviceModel	String	否	无	设备型号

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置App版本

方法名

`setAppVersion`

用法

- 用法如下：

```
module.setAppVersion({
  appVersion: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
appVersion	String	否	无	app版本

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置渠道

方法名

setAppChannel

用法

- 用法如下：

```
module.setAppChannel({
  appChannel: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
appChannel	String	否	无	渠道

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置包名

方法名

setAppPackage

用法

- 用法如下：

```
module.setAppPackage({
  appPackage: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
appPackage	String	否	无	包名

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置标签

方法名

setUserSceneTag

用法

- 用法如下：

```
module.setUserSceneTag({
  //标签只能是数字
  tag: "123"
}, res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
tag	Integer	否	无	标签

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

设置用户ID

- 用法如下：

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
userId	String	否	无	用户id

- 示例

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

模拟Native崩溃

方法名

testNativeCrash

用法

- 用法如下：

```
module.testNativeCrash(res => {
  this.writeLog(JSON.stringify(res))
})
```

- 参数说明
无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败