



作者 19snow93 (/users/b538ca57f640) 2016.09.20 23:17

写了2784字，被18人关注，获得了73个喜欢

(/users/b538ca57f640)

+ 添加关注 (/sign\_in)

# 将安卓app闪退信息保存在本地或服务器的工具类CrashHandler

字数715 阅读142 评论1 喜欢16

在我刚刚实习的时候，有一个测试曾经问我，能不能把安卓app闪退时候的错误信息保存在本地文件夹或者上传到服务器。他说这样做可以把客户反映的闪退问题、而且我们公司不能及时重现BUG的时候去使用。当时，我的回答是否定的，因为我当时并不知道原来安卓早就有UncaughtExceptionHandler可以去全局捕获异常，直到我看了Android开发艺术探索这本书。其实写这篇文章不是为了去展示什么，因为书上、网上都有一模一样的继承UncaughtExceptionHandler写一个工具类。写这篇文章我只是想纠正当年的错误并且可以作为自己项目的工具类去使用，毕竟身为程序猿都不喜欢跟“万恶”的客户和“死神”般的测试有过多的冲突摩擦。当他们发现app有闪退问题的时候，你只需要叫他们把app“临终”前保存在本地的文件打开，这里有几个好处：

1. 可以轻松的摆脱“万恶”的客户和“死神”般的测试的纠缠，有时候客户和测试不在身边，远距离就可以找到问题所在；
2. 你就可以省去很多功夫去重现BUG，可以留下多点时间跟测试的妹子聊天去；
3. 可以把国内各大手机商改过的安卓系统无缘无故会在某些型号的手机上出现闪退情况，捕获记录下来，做好适配工作。

如果有很多app的内容要跟测试妹子深入了解的话，你可以把这篇文章省略掉~

好吧，言归正传，UncaughtExceptionHandler是全局捕获异常的，为app意外中止的提供一些处理的方法。用的时候我们只需要实现UncaughtExceptionHandler这个接口，重写一些方法，就可以做到把异常信息保存在本地或者上传到服务器。

代码如下：

```

public class CrashHandler implements Thread.UncaughtExceptionHandler {
    private static final String TAG = "CrashHandler";
    private static final boolean DEBUG = true;
    private static final String PATH = Environment.getExternalStorageDirectory().getPath() + "/CrashTest/log/";
    private static final String FILE_NAME = "crash";
    private static final String FILE_NAME_SUFFIX = ".txt";
    private static CrashHandler sInstance = new CrashHandler();
    private Thread.UncaughtExceptionHandler mDefaultCrashHandler;
    private Context mContext;
    private CrashHandler(){
    }
    public static CrashHandler getInstance(){
        return sInstance;
    }
    public void init(Context context){
        mDefaultCrashHandler = Thread.getDefaultUncaughtExceptionHandler();
        Thread.setDefaultUncaughtExceptionHandler(this);
        mContext = context.getApplicationContext();
    }
    /**
     * 这个是最关键的函数，当程序中有未被捕获的异常，系统将会自动调用uncaughtException方法
     * thread为出现未捕获异常的线程，ex为未捕获的异常，有了这个ex，我们就可以得到异常信息
     * @param thread
     * @param ex
     */
    @Override
    public void uncaughtException(Thread thread, Throwable ex) {
        try {
            //导出异常信息到SD卡中
            dumpExceptionToSDCard(ex);
            //这里可以上传异常信息到服务器，便于开发人员分析日志从而解决bug
            uploadExceptionToServer();
        } catch (IOException e) {
            e.printStackTrace();
        }
        ex.printStackTrace();
        //如果系统提供默认的异常处理器，则交给系统去结束程序，否则就由自己结束自己
        if(mDefaultCrashHandler != null){
            mDefaultCrashHandler.uncaughtException(thread, ex);
        } else {
            //自己处理
            try {
                //延迟2秒杀进程
                Thread.sleep(2000);
                Toast.makeText(mContext, "程序出错了~", Toast.LENGTH_SHORT).show();
            } catch (InterruptedException e) {
            }
            Log.e(TAG, "error : ", e);
        }
        android.os.Process.killProcess(Process.myPid());
    }
}

private void dumpExceptionToSDCard(Throwable ex) throws IOException{
    //如果SD卡不存在或无法使用，则无法把异常信息写入SD卡

```

```

if(!Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED)){
    if(DEBUG){
        Log.e(TAG, "sdcard unmounted,skip dump exception");
        return;
    }
}
File dir = new File(PATH);
if(!dir.exists()){
    dir.mkdirs();
}
long current = System.currentTimeMillis();
String time = new SimpleDateFormat("yyyy-MM-dd HH:MM:SS").format(new Date(current));
File file = new File(PATH + FILE_NAME + time + FILE_NAME_SUFFIX);
try {
    PrintWriter pw = new PrintWriter(new BufferedWriter(new FileWriter(file)));
    pw.println(time);
    dumpPhoneInfo(pw);
    pw.println();
    ex.printStackTrace(pw);
    pw.close();
    Log.e(TAG, "dump crash info seccess");
}catch (Exception e){
    Log.e(TAG,e.getMessage());
    Log.e(TAG,"dump crash info failed");
}
}

private void dumpPhoneInfo(PrintWriter pw)throws PackageManager.NameNotFoundException{
    PackageManager pm = mContext.getPackageManager();
    PackageInfo pi = pm.getPackageInfo(mContext.getPackageName(),
                                                PackageManager.GET_ACTIVITI
    pw.print("App Version: ");
    pw.print(pi.versionName);
    pw.print('_');
    pw.println(pi.versionCode);
    //Android版本号
    pw.print("OS Version: ");
    pw.print(Build.VERSION.RELEASE);
    pw.print("_");
    pw.print(Build.VERSION.SDK_INT);
    //手机制造商
    pw.print("Vendor: ");
    pw.print(Build.MANUFACTURER);
    //手机型号
    pw.print("Model: ");
    pw.println(Build.MODEL);
    //CPU架构
    pw.print("CPU ABI: ");
    pw.println(Build.CPU_ABI);
}

private void uploadExceptionToServer(){
    //将异常信息发送到服务器
}

```



就是这么简单一个工具类，是不是很简单？好吧下面我们来看看怎么使用~

```
public class CrashApplication extends Application{

    private static CrashApplication sInstance;

    @Override
    public void onCreate() {
        super.onCreate();
        sInstance = this;
        // 在这里为应用设置异常处理，然后程序才能获取未处理的异常
        CrashHandler crashHandler = CrashHandler.getInstance();
        crashHandler.init(this);
    }

    public static CrashApplication getInstance() {
        return sInstance;
    }
}
```

CrashHandler的使用.png

只需要在application获取CrashHandler的单例和初始化就可以做到全局捕获异常了，跟一些第三方的框架初始化没什么区别吧~

最后看一下效果图：



```

private Button mButton;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    initView();
}

private void initView() {
    mButton = (Button)findViewById(R.id.button);
    mButton.setOnClickListener(this);
}

@Override
public void onClick(View v) {
    if(v == mButton){
        //在这里模拟异常抛出情况，人为地抛出一个运行时异常
        throw new RuntimeException("自定义异常：这是自己抛出的异常");
    }
}

```

自己抛出一个异常做测试.png

点击这个按钮，我们就看到app闪退了，然后我们去该保存目录下看看有什么~



2016-09-19 23:09:67

App Version: 1.0\_1

OS Version: 5.1.1\_22Vendor: XiaomiModel: 20148

CPU ABI: armeabi-v7a

```
java.lang.RuntimeException: 自定义异常：这是自己抛
    at com.example.vpcsd.crashhandler.Main
    at android.view.View.performClick(View
    at android.view.View$PerformClick.run
    at android.os.Handler.handleCallback(H
    at android.os.Handler.dispatchMessage
    at android.os.Looper.loop(Looper.java
    at android.app.ActivityThread.main(Ac
    at java.lang.reflect.Method.invoke(Na
    at java.lang.reflect.Method.invoke(Me
    at com.android.internal.os.ZygoteInit
    at com.android.internal.os.ZygoteInit
```

测试结果.png

我们可以看到我们还可以把安卓版本、手机型号、运营商输出来，考虑到国内各大手机商改过的安卓系统无缘无故会在某些型号的手机上出现闪退情况，捕获到这些异常还是很有必要的。

好吧，其实以上的代码很大部分都是来源于Android开发艺术探索这本书的，有兴趣的同学也可以去看一下，我只是想告诉更多还不知道的同学，UncaughtExceptionHandler可以把你的闪退信息保存到本地或者上传到服务器上！

🔗 推荐拓展阅读 (/sign\_in)

© 著作权归作者所有

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作！

¥ 打赏支持

简



🔗 登录 (/sign\_in) 🧑 注册 (/sign\_up)




♡ 喜欢 | 16

分享到微博 分享到微信  
更多分享 ▾

1条评论 ( 按时间正序 · 按时间倒序 · 按喜欢排序 )

✎ 添加新评论 (/sign\_in)

 GageCaiii (/users/e031cc8641a3)  
2楼 · 2016-08-21 00:35 (/p/3f73cfbb6743/comments/4313470#comment-4313470)


其实我想说用腾讯的bugly是不是更方便点呢？


♡ 喜欢(0)


回复

登录后发表评论 (/sign\_in)

被以下专题收入，发现更多相似内容：

 **程序员** (/collection/NEt52a)  
如果你是程序员，或者有一颗喜欢写程序的心，喜欢分享技术干货、项目经验、程序员日常囡事等等，欢迎投稿《程序员》专题。 专题主编：小...  
23809篇文章 (/collection/NEt52a) · 174200人关注

 **Android知识** (/collection/3fde3b545a35)  
分享Android开发的知识，教程，解析，前沿信息，都可以，欢迎大家投稿~ 内容可搞笑，可逗比，另外欢迎申请管理员  
5079篇文章 (/collection/3fde3b545a35) · 21101人关注

 **Android开发** (/collection/d1591c322c89)  
Android开发系列文章，与大家一起交流，欢迎投稿。  
3251篇文章 (/collection/d1591c322c89) · 4127人关注

简



登录 (/sign\_in) 注册 (/sign\_up)