Android使用UncaughtExceptionHandler捕获全局异常

UncaughtExceptionHandler可以用来捕获程序异常，比如NullPointerException空指针异常抛出时，用户没有try catch捕获，那么，Android系统会弹出对话框的“XXX程序异常退出”，给应用的用户体验造成不良影响。为了捕获应用运行时异常并给出友好提示，便可继承UncaughtExceptionHandler类来处理。

1、异常处理类，代码如下：

```java
public class CrashHandler implements UncaughtExceptionHandler {
    private static final String TAG = CrashHandler.class.getSimpleName();

    private static CrashHandler instance; // 单例模式

    private OnBeforeHandleExceptionListener mListener;

    private Context context; // 程序Context对象
    private Thread.UncaughtExceptionHandler defalutHandler; // 系统默认的UncaughtException处理类
    private DateFormat formatter = new SimpleDateFormat(
            "yyyy-MM-dd_HH-mm-ss.SSS", Locale.CHINA);

    private CrashHandler() {

    }

    /**
     * 获取CrashHandler实例
     *
     * @return CrashHandler
     */
    public static CrashHandler getInstance() {
        if (instance == null) {
            synchronized (CrashHandler.class) {
                if (instance == null) {
                    instance = new CrashHandler();
                }
            }
        }

        return instance;
    }

```

```
    /**
     * 异常处理初始化
     *
     * @param context
     */
    public void init(Context context) {
        this.context = context;
        // 获取系统默认的UncaughtException处理器
        defalutHandler = Thread.getDefaultUncaughtExceptionHandler();
        // 设置该CrashHandler为程序的默认处理器
        Thread.setDefaultUncaughtExceptionHandler(this);
    }

    /**
     * 当UncaughtException发生时会转入该函数来处理
     */
    @Override
    public void uncaughtException(Thread thread, Throwable ex) {
        //异常发生时，先给蓝牙服务端发送OK命令
        if (mListener != null) {
            mListener.onBeforeHandleException();
        }

        // 自定义错误处理
        boolean res = handleException(ex);
        if (!res && defalutHandler != null) {
            // 如果用户没有处理则让系统默认的异常处理器来处理
            defalutHandler.uncaughtException(thread, ex);

        } else {
            try {
                Thread.sleep(3000);
            } catch (InterruptedException e) {
                Log.e(TAG, "error : ", e);
            }
            // 退出程序
            android.os.Process.killProcess(android.os.Process.myPid());
            System.exit(1);
        }
    }

    /**
     * 自定义错误处理,收集错误信息 发送错误报告等操作均在此完成.
     *
     * @param ex
     * @return true:如果处理了该异常信息;否则返回false.
     */
    private boolean handleException(final Throwable ex) {
        if (ex == null) {
            return false;
        }

        new Thread() {

            @Override
```

```
            public void run() {
                Looper.prepare();

                ex.printStackTrace();
                String err = "[" + ex.getMessage() + "]";
                Toast.makeText(context, "程序出现异常." + err, Toast.LENGTH_LONG)
                        .show();

                Looper.loop();
            }

        }.start();

        // 收集设备参数信息 \日志信息
        String errInfo = collectDeviceInfo(context, ex);
        // 保存日志文件
        saveCrashInfo2File(errInfo);
        return true;
}
```

2、应用绑定异常处理方法：

在Application或者Activity的onCreate方法中加入以下两句调用即可：

```
[java]
<span style="font-
family:Courier New;">CrashHandler crashHandler = CrashHandler.getInstance();
crashHandler.init(getApplicationContext());</span>
```