<http://www.jianshu.com/p/216b03c22bb8>

<https://www.liaohuqiu.net/cn/posts/leak-canary-read-me/>

<http://hanhailong.com/2015/12/27/Android%E6%80%A7%E8%83%BD%E4%BC%98%E5%8C%96%E4%B9%8B%E5%B8%B8%E8%A7%81%E7%9A%84%E5%86%85%E5%AD%98%E6%B3%84%E6%BC%8F/>

<https://gist.github.com/pyricau/4726389fd64f3b7c6f32>

**LeakCanary 工作机制**：

RefWatcher.watch() 创建一个 KeyedWeakReference 到要被监控的对象。

然后在后台线程检查引用是否被清除，如果没有，调用GC。如果引用还是未被清除，把 heap 内存 dump 到 APP 对应的文件系统中的一个 .hprof 文件中。

在另外一个进程中的 HeapAnalyzerService 有一个 HeapAnalyzer 使用HAHA 解析这个文件。得益于唯一的 reference key, HeapAnalyzer 找到 KeyedWeakReference，定位内存泄露。

HeapAnalyzer 计算 到 GC roots 的最短强引用路径，并确定是否是泄露。如果是的话，建立导致泄露的引用链。

引用链传递到 APP 进程中的 DisplayLeakService， 并以通知的形式展示出来。