infer 代码静态扫描工具接入

一，背景

在我们的Xcode工程中，有很多没有引起重视的警告，还有一些判空逻辑的缺失以及一些内存泄漏的问题，基于这些问题，我们出现过几次线上问题，而这些问题，我们在快速的业务开发过程中，很容易忽略，提交的 merge request 也看不到警告，就这样让有问题的代码上线了。

随着业务越来越多，开发人员越来越多的情况下，迫切需要一种自动化代码检测工具来做一些代码审查的兜底工作，让我们能减少一些低级错误，减少线上问题的发生，经过对比 OCLint 和 Facebook 的 infer ，最终还是觉得infer 更适合我们的项目，它分析问题的能力更精准，也能增量分析

二，infer 介绍及安装

infer 是 facebook 开源的代码静态分析工具，它可以检查出 OC，Java,C的潜在问题，让一些低级编码问题无处可藏，减少人为的失误，支持的检测类型如下：

- DEAD_STORE，创建对象，而并没有使用
- DIRECT_ATOMIC_PROPERTY_ACCESS，在代码中使用了使用了一个atomic的成员变量
- BAD_POINTER_COMPARISON，错误指向比较
- NULL_DEREFERENCE，空指针的情况
- PARAMETER_NOT_NULL_CHECKED，参数非空判断
- MEMORY_LEAK，内存泄漏
- ASSIGN_POINTER_WARNING，这个属性被判断为assign，需要将其修改为weak或者strong
- REGISTERED_OBSERVER_BEING_DEALLOCATED，注册的通知没有被销毁
- RESOURCE_LEAK，资源泄漏
- STRONG_DELEGATE_WARNING，代理属性设置为strong
- ...

其中 DEAD_STORE，NULL_DEREFERENCE，MEMORY_LEAK ，ASSIGN_POINTER_WARNING 几个最容易出现，也是影响比较大的几个 issues;

infer的安装很简单，使用 brew 就可以安装

- brew install infer

2.1 infer 的常用命令

infer 的代码分析流程主要分为两步，第一步是代码的编译，第二步是代码的分析；

- infer capture ,代码编译过程，Xcode 工程需要跟 xcodebuild 命令来完成编译
- infer analyze ,代码分析过程，可以指定只分析部分文件，而不用全部分析，避免相同文件的重复分析
- infer run ，上面两个命令的合成

一般来说，进入到工程目录，执行 infer run 就可以完成整个分析过程了

示例：infer run --skip-analysis-in-path Pods --xcodebuild -workspace (workspace name) -scheme (scheme name) --configuration Debug -sdk iphoneos

--skip-analysis-in-path :添加分析的忽略目录

注：infer 官网命令说明

三，项目接入

3.1 采用的 infer 命令

由于贪吃蛇项目太大了，分析的文件过多，采用上面一步的命令得到的分析结果太多，有效的分析结果占比太低，所以采用三步来完成，指定需要分析的文件，如下：

- xcodebuild -workspace $myworkspace -scheme $myscheme -configuration Debug -sdk iphoneos COMPILER_INDEX_STORE_ENABLE=NO | tee xcodebuild.log ,生成编译log
- xcpretty -r json-compilation-database -o compile_commands.json < xcodebuild.log > /dev/null,根据编译日志生成编译数据compile_commands.json文件
- infer run --no-xcpretty --keep-going \

--skip-analysis-in-path Pods \ ##指定跳过的分析路径

--changed-files-index git_change_files.txt \ ##指定分析的文件列表

--compilation-database-escaped compile_commands.json \ ##指定编译数据json文件

--disable-issue-type DIRECT_ATOMIC_PROPERTY_ACCESS \ ##忽略属性atomic声明issue

--disable-issue-type MULTIPLE_WEAKSELF \ ##忽略weak-strong未使用的警告

--disable-issue-type UNINITIALIZED_VALUE \ ##忽略变量未初始化的警告

--disable-issue-type POINTER_TO_INTEGRAL_IMPLICIT_CAST \ ##忽略变量数据类型静默转化的警告

--disable-issue-type BAD_POINTER_COMPARISON ##忽略不同类型判断的警告

注：详见项目下的 infer.sh

3.2 增量分析

本来 infer 是自动增量分析的，但是通过贪吃蛇工程的测试发现，增量编译会失败，而且增量编译的效果也包含很多，所以采用 --changed-files-index 参数git 对比两次分析的commitId来获取文件改动，

git_change_files.txt 是根据相邻两次分析的文件改动来修改动态生成的，具体步骤如下：

- git diff $new_commit_id $last_commit_id --name-status >> git_raw_change_files.txt
- 然后处理 git_raw_change_files.txt ，去掉其中的图片等非代码文件生成git_change_files.txt

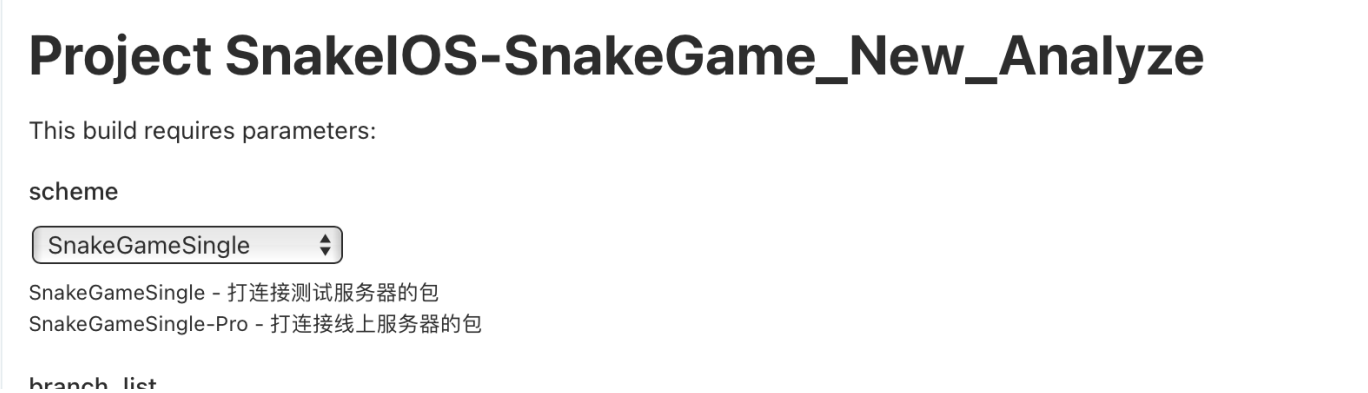
```
function updateGitChangeFilesTxt () {

    last_commit_id= $(sed -n '1p' git_raw_change_files.txt)
    echo $last_commit_id
    new_commit_id=$(git rev-parse --short HEAD)
    echo $new_commit_id
    echo $new_commit_id > ./git_raw_change_files.txt
    git diff $new_commit_id $last_commit_id --name-status >> ./git_raw_change_files.txt
    file_names=$(cat ./git_raw_change_files.txt)
    is_first=false
    git_change_file_count=0
    for line in $file_names
    do
        if echo "$line" | grep -q -E '\.h$' || echo "$line" | grep -q -E '\.m$'
        then
            if [[ $is_first = 'false' ]]
            then
                echo $line >./git_change_files.txt
                is_first='true'
            else
```

```
                git_change_file_count= expr $git_change_file_count + 1
                echo $line >>./git_change_files.txt
            fi
        fi
    done
    echo '文件修改的个数为'$git_change_file_count
    if [[ $git_change_file_count -le 2 ]]
    then
        exit
    fi
```

3.3 Jenkins 集成

工程名称：SnakeIOS-SnakeGame_New_Analyze

选择分支，默认是dev，有特殊情况可以修改分支



四，使用

编译完成可以点击这里的 report.txt 查看分析的结果

```
Found 156 issues
                                                Issue Type(ISSUED_TYPE_ID): #
```

```
                                Null Dereference(NULL_DEREFERENCE): 70
    Unsafe Call To Optional Method(UNSAFE_CALL_TO_OPTIONAL_METHOD): 20
            Parameter Not Null Checked(PARAMETER_NOT_NULL_CHECKED): 14
                  Strong Delegate Warning(STRONG_DELEGATE_WARNING): 12
                         Captured strongSelf(CAPTURED_STRONG_SELF): 12
                    Assign Pointer Warning(ASSIGN_POINTER_WARNING): 11
                                          Dead Store(DEAD_STORE): 6
                          Mixed Self WeakSelf(MIXED_SELF_WEAKSELF): 5
   Pointer To const Objective-C Class(POINTER_TO_CONST_OBJC_CLASS): 3
        Weak Self In No Escape Block(WEAK_SELF_IN_NO_ESCAPE_BLOCK): 2
                    StrongSelf Not Checked(STRONG_SELF_NOT_CHECKED): 1
```

Origin Link: https://wepie.yuque.com/tcsdzz/ios_team/uiu6ud