

# Статический анализатор clang

- clang -cc1 -analyze -analyzer-checker=***package***  
***file***
- clang -analyze -Xanalyzer  
-analyzer-checker=***package file***
- clang -cc1 -analyze -analyzer-checker-help
- clang -cc1 -help | grep analyzer
- clang -cc1 -analyze -analyzer-checker=***core***  
***file.c*** -o report

# Статический анализатор clang.

## Пакеты:

Alpha		
core		
debug		
llvm		
osx		
security		
unix		
cplusplus		

# Статический анализатор clang для проектов

- scan-build **команда\_сборки**
- scan-build gcc -c file.c -o file
- scan-view **каталог\_с\_отчетами**

## **Пример:**

- mkdir build && cd build
- scan-build ../configure -prefix=\$(pwd)/../install
- scan-build make

# Расширение статического анализатора clang

- Определение и реализация подкласса Checker
- Регистрация в пакете alpha
- Сборка и тестирование нашего средства проверки (см. пример ReactorChecker.cpp)

# Инструменты Clang и LibTooling

- Установить Clang Extra Tools
- Создать базу данных команд компиляции
  - mkdir build && cd build
  - cmake
    - DCMAKE\_EXPORT\_COMPILE\_COMMANDS=ON ..***
  - ln -s compile\_commands.json ..
- ***или***
  - libtool test.c -- -include  
прочие\_параметры\_компиляции

# Инструменты Clang и LibTooling

- ***Clang-tidy*** – проверки на нарушение стандартов оформления
- ***Clang-modernize*** – адаптация старого кода под новые стандарты, например C++11
- ***Clang-apply-replacements*** – в помощь Clang-modernize для больших проектов
- ***Clang-format*** – форматирование C/C++ кода
- ***Modularize*** и ***module-map-checker*** – решение проблем внедрения модулей (\*.cpp), анализ заголовочный файлов, глобальных переменных
- ***Pp-trace*** – вывод трассы работы препроцессора
- ***Clang-query*** и ***clang-check*** – работа с AST-деревом
- ***Remove-cstr-calls*** – удаление `.c_str()` вызовов
- ***Создание собственного инструмента*** рефакторинга, например переименование переменных (см. `IzzyRefactor.cpp`)

# Инструменты Clang и LibTooling

*libtool* [параметры] file0, ..., fileN – параметры\_компиляции

- *clang-tidy -checks="llvm-\*" test.c --*
- *clang-modernize -loop-convert -serialize-replacements test.c --*  
– создаются (по умолчанию в текущей директории yaml файлы), чтобы применить изменения:
- *clang-apply-replacements ./*
- *clang-format -style=llvm|google|chromium|mozilla|webkit test.c --*
- создаем файл list.txt с содержанием вида:

*file.c*

*file.h*

далее

*modularize list.txt*

*module-map-checker module.modulemap*

- *pp-trace test.c –*