- 基本流程 示： App.c:文件名/ **main**:函数名/ *文字*:函数说明/ (文字):解释
- AppCrawler.scala
  - **getGlobalEncoding** 方法、**setGlobalEncoding** 方法：略。
  - **main** 方法：所有执行都从 main 函数进去。会先解析参数，内容很多。所有逻辑都在解析参数中，参数类型已经在 **createParser** 的 help("help") text 中给出。调用 **parseParams** 方法。
  - **createParser** 方法：略。
  - **parseParams** 方法：解析参数。根据参数类型，执行不同功能，包括设置基础的 log 等，最后会打印 config。命令行中可以传递 capability 对已有配置文件进行覆盖。接着是各种参数解析。直到最后来执行，一种是遍历一种是报告。

    其中一个命令模式是生成 demo 文件，在 if(config.demo)段中，生成后自动退出。其余功能是遍历，包括未完成的（基于 PO 的自动）生成测试用例源代码的模板 if(config.template)代码块、关于 diff 逻辑的 if(config.candidate)代码块（其功能为如若不为空则 diff）、没有用到的可实现报告重复生成的功能（把所有 log 删掉，只要有 elements.yml 文件和 store 存在，会重新根据文件的点击顺序再次重新生成报告），其可以用来针对修改报告中断言后，预期某些步骤报错后可以重新生成报告。最终调用 **startCrawl**。
  - **parsePath** 方法：略。
  - **startCrawl** 方法：初始化 Crawler 对象，从配置文件里面加载配置，并 start。
  - **addLogFile** 方法：略。
- Crawler.scala
  - **loadPlugins** 方法、**loadConf** 方法：略。
  - **start** 方法：所有 log 都在 AppCrawler 里，通过 *log.addAppender(AppCrawler.fileAppender)*把当前的 log 增加一个输入到总体 log 里面。debug 数据并打印配置文件方便调试。