Java代码度量分析工具：Designite简介

2019-03-24

前言

在Java面向对象课程的学习过程中，我们需要使用度量工具来分析自己程序的代码结构。此类的度量工具有许多，或以插件形式存在于各个IDE中，或以.jar包的形式供用户使用。在这里，笔者向大家简单的介绍一款免费的代码度量分析工具：Designite，对官网上的部分内容进行汉化搬运，并对一些度量条目做出解释。

一、Designite的下载与运行

Designite是一款程序设计的质量评估工具。这款工具可以用于分析C#和Java的代码，并且识别其中存在的质量问题。Designite会检测程序的架构(architecture)，设计(design)和代码异味(code smell)并且给出详细的度量分析(metrics analysis）。

Designite官网：http://www.designite-tools.com/

Designite for Java：http://www.designite-tools.com/DesigniteJava/

进入官网，下载会得到一个jar包，默认名为DesigniteJava.jar。将其放在方便使用的位置，打开命令行窗口，输入以下代码即可运行：

```
java -jar DesigniteJava.jar -i <源文件路径> -o <分析数据输出路径>
```

输出之后，会得到以下五个文件。

designCodeSmells和implementationCodeSmells中存储着在设计和执行阶段检测到的**代码异味(code smell)**。这标志着你的程序中存在一些设计上的不规范，如过多幻数(Magic Number)，模块之间的循环依赖(Cyclic Dependent)等。

methodMetrics和typeMetrics中存储着类和方法的一些**度量指标(metrics)**，这些度量指标有助于我们分析程序的整体结构，并且找出问题所在。

二、度量指标介绍

以下内容翻译自官网，简单加上了一些个人理解。

- **LOC** (Lines Of Code – at method and class granularity)
  代码行数，可以看到你的方法和类写了多少行。

- **CC** (Cyclomatic Complexity – Method)
  圈复杂度，用于衡量一个模块判定结构的复杂程度，圈复杂度越大说明程序代码质量低，且难以测试和维护。

- **PC** (Parameter Count – Method)
  方法中传入的参数个数。

- **NOF** (Number of Fields – Class)
  类的字段个数。

- **NOPF** (Number of Public Fields – Class)
  类的公共(public)型字段个数。

- **NOM** (Number of Methods – Class)
  类的方法个数。

- **NOPM** (Number of Public Methods – Class)
  类的(public)型方法个数。

- **WMC** (Weighted Methods per Class – Class)
  类的加权方法个数。具体加权算法怎么算，笔者不太清楚。

- **NC** (Number of Children – Class)
  类的子类个数。

- **DIT** (Depth of Inheritance Tree – Class)
  类所在的继承树深度。

- **LCOM** (Lack of Cohesion in Methods – Class)
  方法的内聚缺乏度。值越大，说明类内聚合度越小。

- **FANIN** (Fan-in – Class)
  类的扇入。扇入表示调用该模块的上级模块的个数，扇入越大，表示该模块的复用性好。

- **FANOUT** (Fan-out – Class)
  类的扇出。扇出表示该模块直接调用的下级模块的个数，扇出过大表明模块复杂度高，但扇出过小也不好。
  设计要求一般是**高内聚低耦合**，即LCOM值要小，FANIN值要大，FANOUT值要合理。