**APT简介**

赵曈

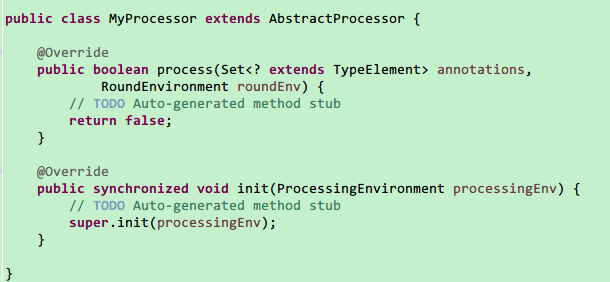
**一、APT是什么**

APT，即Annotation Processing Tool注解处理器。是由JDK自带的（1.6及以上）、供用户编写实现自定义Annotation（Java注解）处理逻辑的一系列API及编译工具。

Annotation相信大家不会陌生，在我们编程中，像@Override这类就是一种Annotation。在Java世界中，Annotation有作用域（字段、函数、类）和作用时间（源码时、编译时、运行时）区别，在此就不详细展开了。一般来说，Annotation都伴随着Java反射技术一同使用，可以达到用户使用简单的Annotation标注一个Java Bean，通过简单调用，相关库在程序运行时将被标注的Java Bean转换为格式化数据。市面上有很多流行的库，比如说Gson、FastJson、OrmLite等等都使用了这种技术。

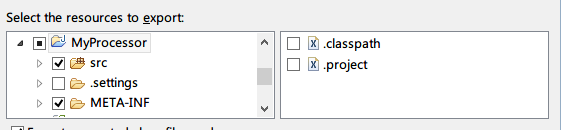
而本文主角APT与上面所说的技术有一点最大区别，即通过使用APT，程序员可以在程序编译前，在每个Java类中检测符合自定义规则的元素，并且执行自定义的逻辑，例如自动生成新的Java类文件。APT动作完成之后，程序再继续正常的编译行为。

**二、APT如何使用**

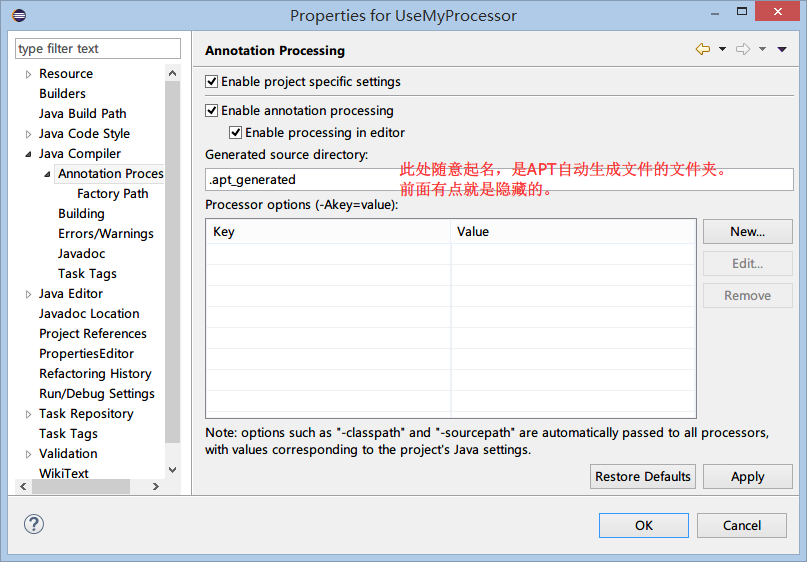
STEP 1.新建工程，新建一个Java类，就叫MyProcessor好了。MyProcessor需要继承javax.annotation.processing.AbstractProcessor，覆写一些方法，如图：

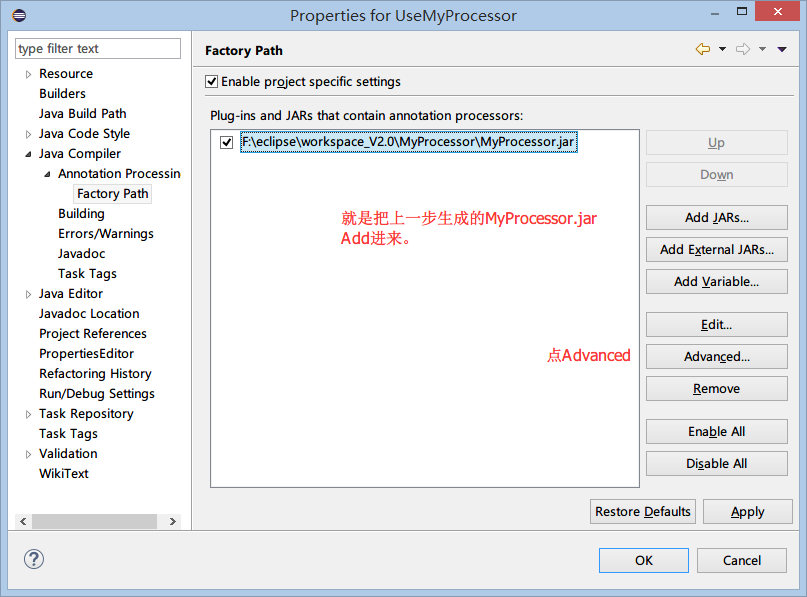
process方法就是APT的执行点，我们对Java类的解析和处理就在此函数中进行。

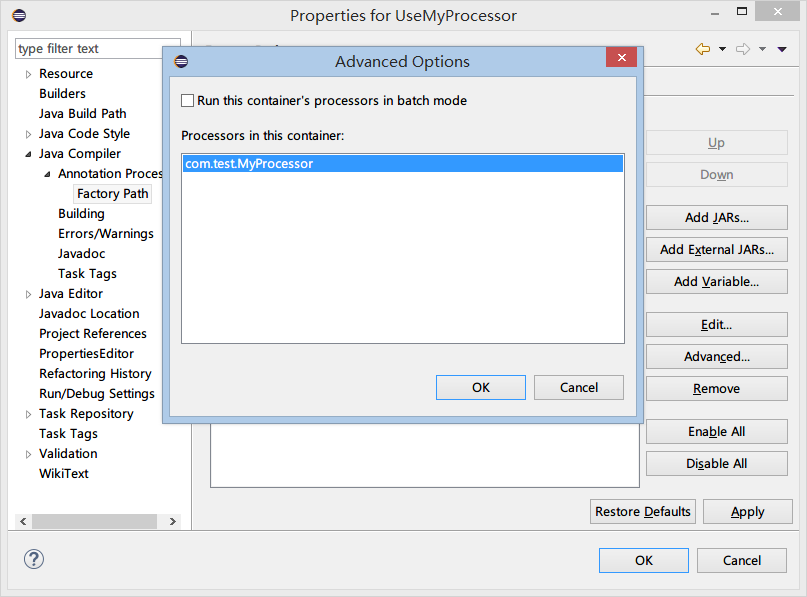
STEP 2.在工程中创建新文件夹，名为”META-INF”，二级文件夹”services”，在其中创建文件，命名为”javax.annotation.processing.Processor”，文件中写上MyProcessor类的完整名称，如”com.test.MyProcessor”

STEP 3.将此工程打包为jar包，勾选如图：

STEP 4.建立另一个测试工程，右键工程选择如下：







之后OKOKYESYES搞定。

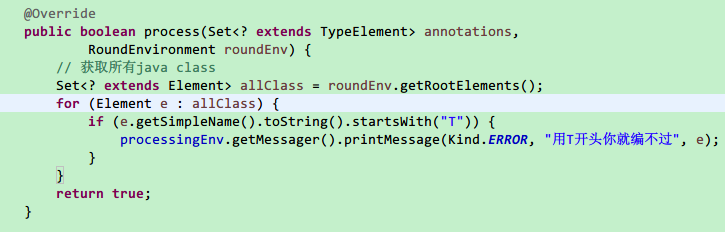
至此我们完成了自己的注解处理器编写和使用，当然，你会发现完全没变化，那是因为我们的process函数里面啥都没写，下一节详细说下。

**三、APT能用来干什么**

STEP 1. 让我们从简单的开始，现在，我想规定所有在我手下的程序员们给类起名字都不能用T开头。那么声明注解处理器要处理的文件、支持的最高Java代码版本。后续处理时，可以根据实际需要，在@SupportedAnnotationTypes的value中赋值你想进行处理的Annotation完整className。

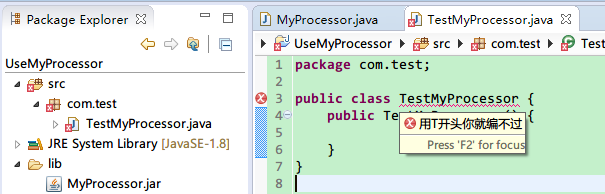


STEP 2.改造MyProcessor类。Element有许多实现，这些实现表示了字段、函数、类等不同类型，不同类型Element中可获取到不同的Kind，可供用来进行一些判断，如是否私有函数，字段数据是什么类型等等。



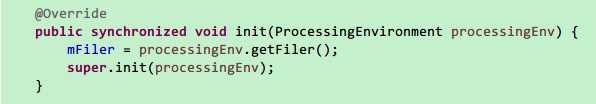
STEP 3.生成jar包，让所有程序员按照上面的步骤导入。

STEP 4.程序员不信邪，写了一个类。看他编不过了吧。当然，我们也可以不提示Error，还有好几种类型的提示可供选择。



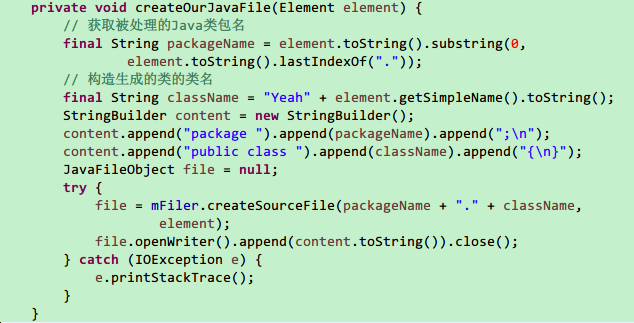
STEP 5.现在换个花样，给所有用T开头的类，在其所属包下生成一个带Yeah前缀的Java文件。

首先获取到文件操作类Filer

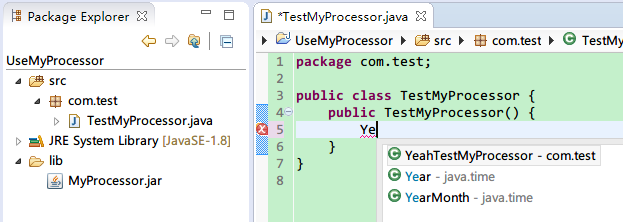


其次写处理的逻辑



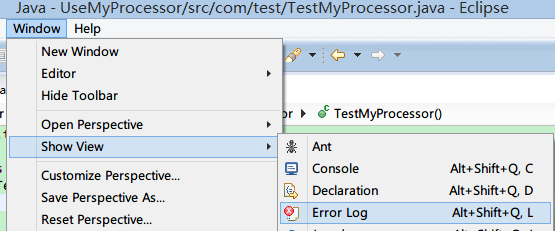


最后生成Jar包，更新。友情提示，更新Jar包操作不要在Eclipse里面做，会报无法覆盖。请直接在操作系统下，把生成的Jar包直接复制粘贴到相应工程目录下。之后刷新/Clean让工程重新编译即可。可以看到已经有这个类了。



为什么我们在左边看不到呢？回忆下第二节中，设置中有一项指定自动生成的路径，前面默认是带点，就是隐藏的了。如果我们想看到的话改改就行。



 STEP 6.注解处理器中如何Debug。APT处理时在正常编译前执行的，所以我们没法像普通程序一样在Log里面看到错误和打印。如果需要加打印，建议使用上面介绍的printMessage方法，直接把要加的打印显示在相关Elment上，也很直观。如果想看到APT运行时的错误，需要打开Window-Show View-Error Log，如图

**四、APT工程示例**

我与许绍秋同学联手制作了一款Android SQLite ORM框架。参考了世面上比较流行的一些ORM框架，比如AFinal，xUtils，DBExecutor等等。我们框架的优势就在于使用了APT技术，而其余框架使用运行时反射。众所周知反射是需要额外时间的，会导致效率变低。我们在918上面跑过测试，10w行数据的增删改查，速度比上述框架快很多。当然，我们认为易用性方面也有优势，具体就不吹了。此框架目前在我们自己的应用，如全网搜索、EPGServer、呼吧3.0等应用中都有所使用。欢迎大家围观使用吐槽。Github地址：<https://github.com/zxfrdas/SimpleDAO>

个人认为APT可以做的事情还有很多，比如我们可以用来生成findViewById的代码，或者对类、函数的命名、属性做规定。期待大家更好的点子。

Have Fun!