说明

包以及详细的类（详情见doc部分中的文档）

- agent
  - AppInfo
    包含黑名单，白名单，App实例名以及相关的获取，添加，匹配操作方法。
  - Interceptor
    所有插入到字节码中的方法，都在其中
  - Monitor
    代理启动类
  - XMLReader
    xml配置文件读取，读取后将数据加入到AppInfo对应的白名单和黑名单中

- record
  - Collector
    数据本地缓存以及写入Codis的线程类
  - Record
    写入Codis的日志类

- traceClass
  - Trace
    ThreadLocal中所用的记录类

- transation
  - ams(包)
    MonitorClassAdapter
    AMS访问类的时候调用
    MonitorMethodAdapter
    AMS访问方法的时候调用
  - ClassTransformer
    只要不是核心API加载的时候，类都会通过这里，进行ClassName匹配拦截

- util
  - SpanUtils
    版本号工具
  - UID
    唯一标识符生成器

使用

1. Jar包方面
   使用IDE自动编译获取Jar包
   依赖的jar包已经放在了压缩包中，
   压缩包中包含最新版本jar包一份，以及一份配置文件示例，以及一份使用javaDoc生成的实例
2. 需要的配置文件的XML格式方面，名称为config.xml
   参考如下格式

```
<?xml version="1.0" encoding="UTF-8"?>
<interceptor>
    <Ignore>com.danlu.dlmonitor.vo</Ignore>
    <Add>com.danlu</Add>
</interceptor>
```

其中add标签标识需要拦截的包名，ignore标识不需要拦截的，上面例子则表明，屏蔽了com.danlu.dlmonitor.vo下面所有类
对于com.danlu下其中所有文件进行拦截
默认对于包名中包含persist.entity（即entity类）以及dlcommon还有dlhttpx中的类进行了屏蔽
推荐只对于相应的工程的包进行拦截，例如在dlmonitor工程下，只对com.danlu.dlmonitor进行添加，以减小消耗
推荐尽量屏蔽POJO类，因为其中seter和getter方法会打出大量无用日志

编辑解压后目录中xml文件，
设置虚拟机启动参数如下

```
-javaagent:C:\Users\y1247\Desktop\lib\agent.jar
-noverify
-Dagent.appName=AppAAA
-Dagent.dir=C:\Users\y1247\Desktop\lib\
```

其中javaagent为jar包目录
-Dagent.appName为当前App的实例名称
-Dagent.dir为xml目录。

技术细节

拦截部分

1. 启动时 使用代理添加了ClassTransFormer类来对每个类的字节码进行处理。
2. 在加载时，首先使用传入加载的类的二进制流来创建CtMethod(*Javassist方式*)或者ClassReader(*ASM方式*)，并使用其进行代码插入

3. 插入后每次请求都是从doDispatch开始记录，ThreadLocal中保存记录类，每次方法调用会对当前线程的ThreadLocal进行处理，并且在调用方法和方法返回时分别记录一条日志。
4. 在跨Http调用方面，会在HttpX中对请求头加入相关的追踪信息，收到http请求时会判断是否携带所有需要的信息，如果没有则认为是一条来自用户的http请求，直接进一次新的http调用记录，如果有，则将信息方法记录，计入对应调用链
5. 在RPC调用时，与http相似，会在传递过程中，将相应的信息传递方法传递报文中，在到达后从报文中取出，进行记录

写入Codis部分

1. 本地存在2个实际队列，即两个缓冲区，2个指针，分别指向写入队列和读取队列
2. 会在写入线程完成当前写入队列中所有数据的写入工作后进行缓冲区切换，即，交换写入指针与读取指针所指向的实际队列。