**Members：**

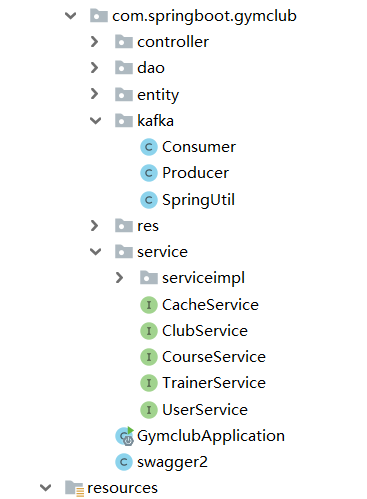
16301045 彭浩伟

16301047 冉旭松

16301050 薛飞跃

**项目结构：**

代码结构：

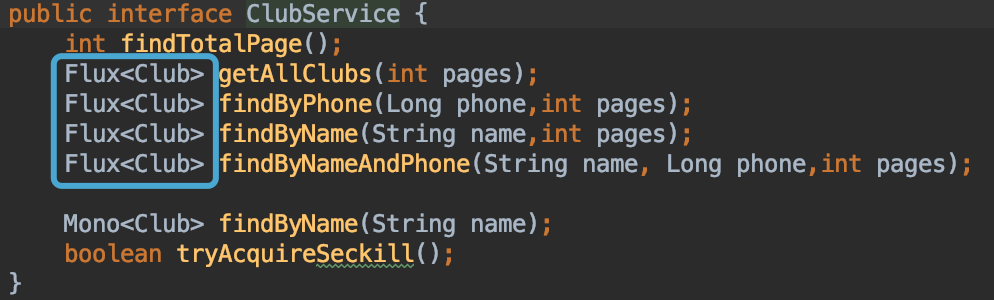


**webFlux替换springMVC：**Webflux模式替换了旧的Servlet线程模型。用少量的线程处理request和response io操作，而业务交给响应式编程框架处理，用户可以将业务中阻塞的操作提交到响应式框架的work线程中执行，而不阻塞的操作依然可以在Loop线程中进行处理，大大提高了Loop线程的利用率。

而由于本项目之前就一直是使用的springboot，所以修改成webflux比从springmvc重构要简单。

**代码及功能展示：**

* 由于是重构项目，功能与之前一致，这里不做多余的功能介绍



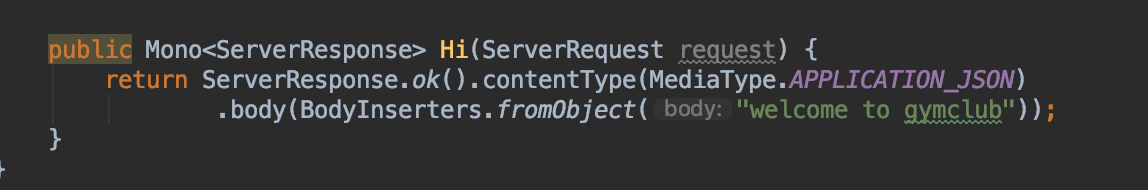
在接口层，我们将List类型全部替换成了Flux类型，将原来的单个User类型转换成Mono，这是webflux的特性。

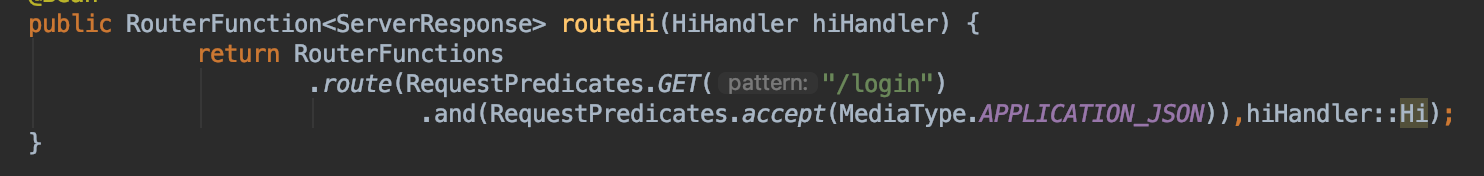


在实现层，我们将原来的方法重构，然后将返回值框上flux指定的方法。

在引用层我们就将原本的变量声明修改成对应的Flux或者Mono类型的变量或者返回值。

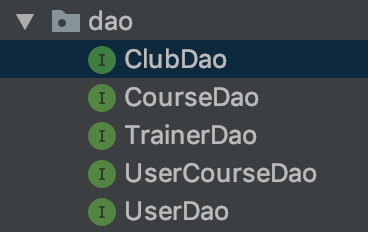
* 接下来是函数式编程

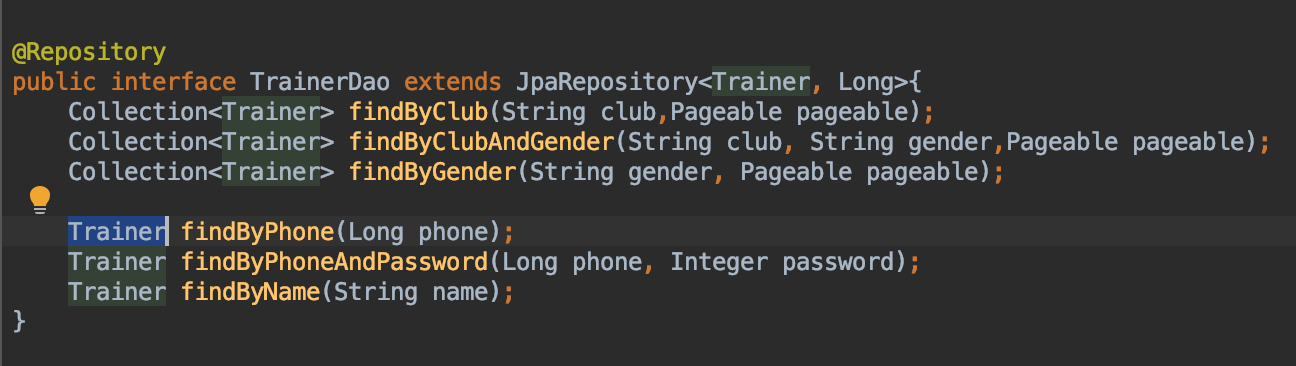


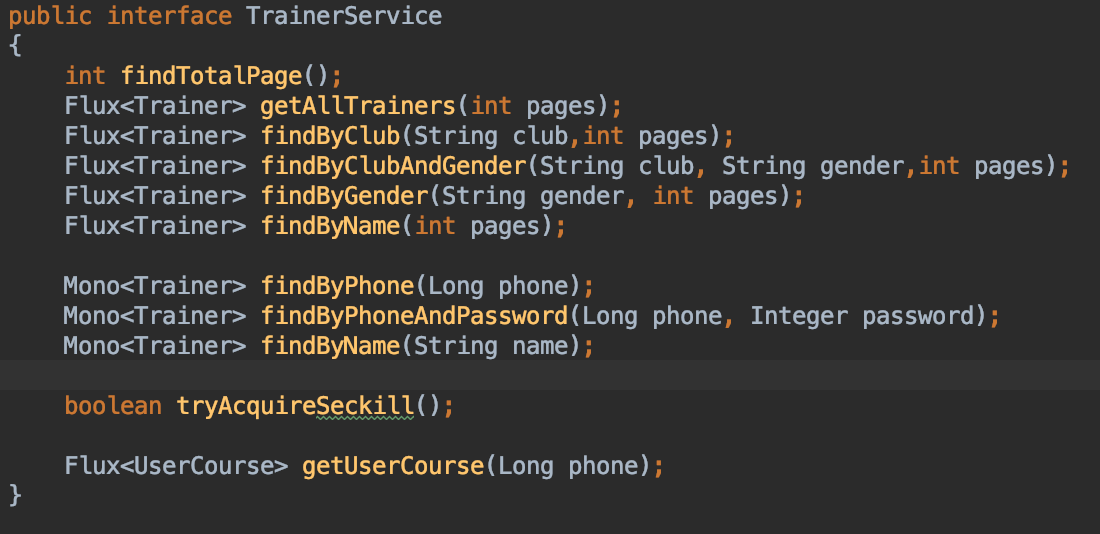


函数式编程是种编程方式，它将电脑运算视为函数的计算。函数编程语言最重要的基础是lambda calculus，而且λ演算的函数可以接受函数当作输入（参数）和输出（返回值)。我们将以前的mvc注解模式，也就是我们最常使用的一种模式，转换成函数式编程模式，我们可以从中看见，虽然webflux兼容注解模式，但是webflux的特色就是函数式编程。

* 持久性是数据的一个属性，其确保即使是在应用的生命周期之外数据也是可用的。对于像Java这样的面向对象语言来说，持久性确保了即使是在创建对象的应用停止执行之后对象的状态仍是可访问的。







* Swagger由于在重构为webflux之后，并不兼容，也就是说并不能使用swagger了。

