# 1.开启基于注解的缓存 ： @EnableCaching



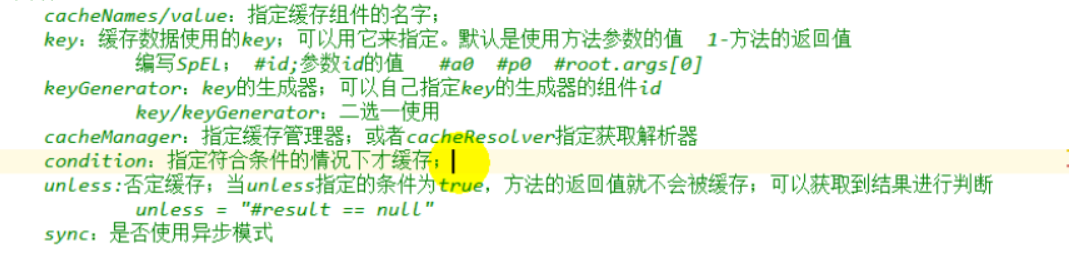
1. 标注缓存注解

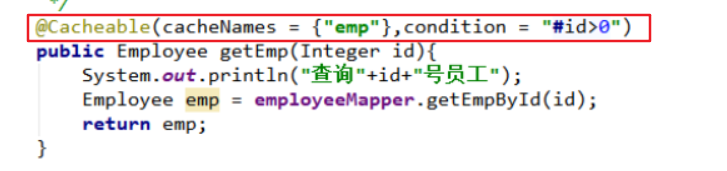
## @Cachable：将运行结果进行缓存。

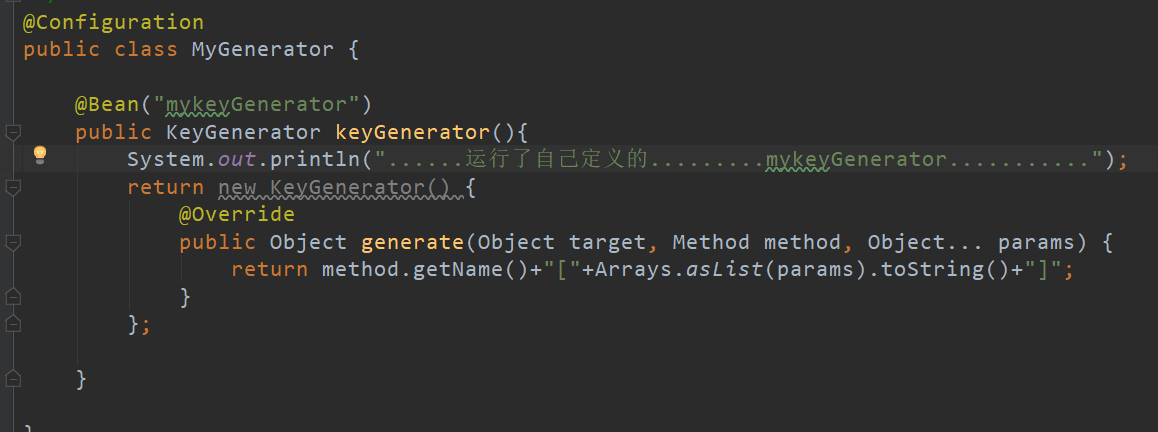
@Cachable：将方法的运行结果进行缓存，以后再要相同的数据，直接从缓存中获取，不用再调取方法。

CacheManage管理多个Cache组件，对缓存的真正CRUD操作在Cache组件中，每一个缓存组件有自己唯一一个名字。

几个属性：







原理：

1、自动配置类；CacheAutoConfiguration

2、缓存的配置类

org.springframework.boot.autoconfigure.cache.GenericCacheConfiguration

org.springframework.boot.autoconfigure.cache.JCacheCacheConfiguration

org.springframework.boot.autoconfigure.cache.EhCacheCacheConfiguration

org.springframework.boot.autoconfigure.cache.HazelcastCacheConfiguration

org.springframework.boot.autoconfigure.cache.InfinispanCacheConfiguration

org.springframework.boot.autoconfigure.cache.CouchbaseCacheConfiguration

org.springframework.boot.autoconfigure.cache.RedisCacheConfiguration

org.springframework.boot.autoconfigure.cache.CaffeineCacheConfiguration

org.springframework.boot.autoconfigure.cache.GuavaCacheConfiguration

org.springframework.boot.autoconfigure.cache.SimpleCacheConfiguration【默认】

org.springframework.boot.autoconfigure.cache.NoOpCacheConfiguration

3、哪个配置类默认生效：SimpleCacheConfiguration；

4、给容器中注册了一个CacheManager：ConcurrentMapCacheManager

5、可以获取和创建ConcurrentMapCache类型的缓存组件；他的作用将数据保存在ConcurrentMap中；

运行流程：@Cacheable：

1、方法运行之前，先去查询Cache（缓存组件），按照cacheNames指定的名字获取；（CacheManager先获取相应的缓存），第一次获取缓存如果没有Cache组件会自动创建。

2、去Cache中查找缓存的内容，使用一个key，默认就是方法的参数；

key是按照某种策略生成的；默认是使用keyGenerator生成的，默认使用SimpleKeyGenerator生成key；

SimpleKeyGenerator生成key的默认策略；

如果没有参数；key=new SimpleKey()；

如果有一个参数：key=参数的值

如果有多个参数：key=new SimpleKey(params)；

3、没有查到缓存就调用目标方法；

4、将目标方法返回的结果，放进缓存中

@Cacheable标注的方法执行之前先来检查缓存中有没有这个数据，默认按照参数的值作为key去查询缓存，

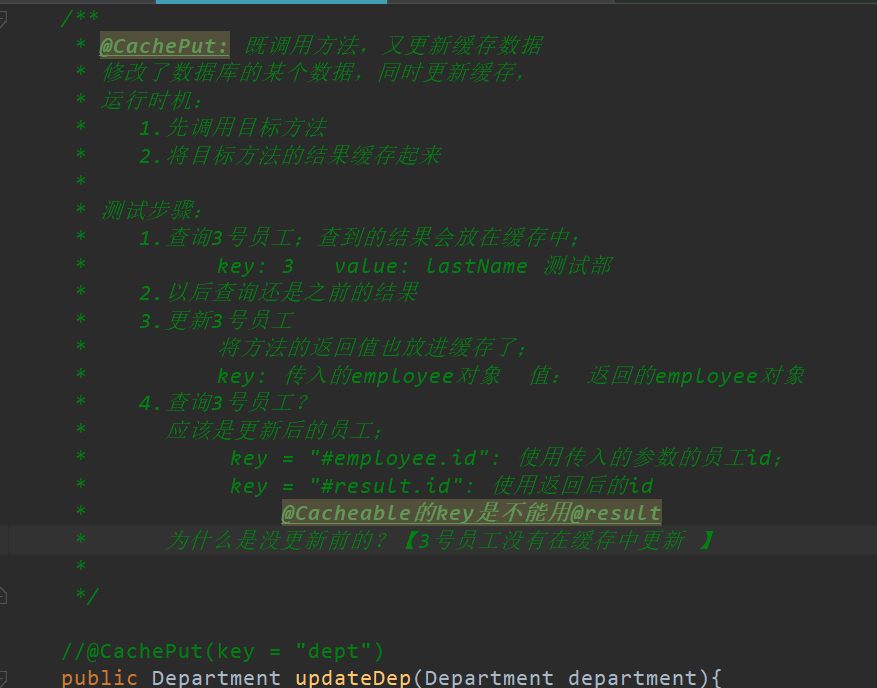
如果没有就运行方法并将结果放入缓存；以后再来调用就可以直接使用缓存中的数据；

核心：

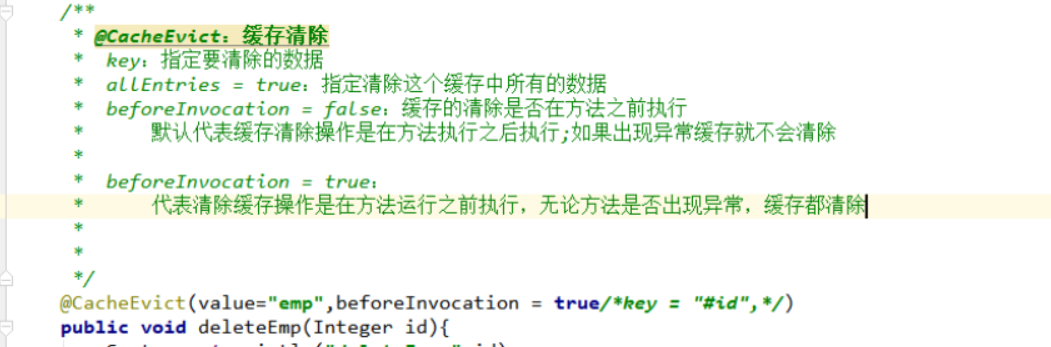
1）、使用CacheManager【ConcurrentMapCacheManager】按照名字得到Cache【ConcurrentMapCache】组件

2）、key使用keyGenerator生成的，默认是SimpleKeyGenerator

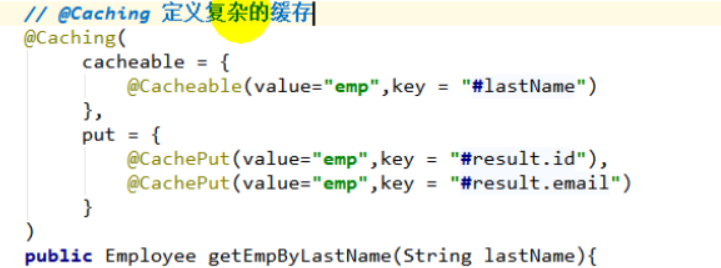
## @cachePut：即调用方法，又更新缓存数据



## @CacheEvict：清除缓存



## @Caching: 复杂的缓存



还可以在类上添加：在上面指定了，下面就不用再指定了，就可以在emp缓存中进行查

## @Cacheconfig：

