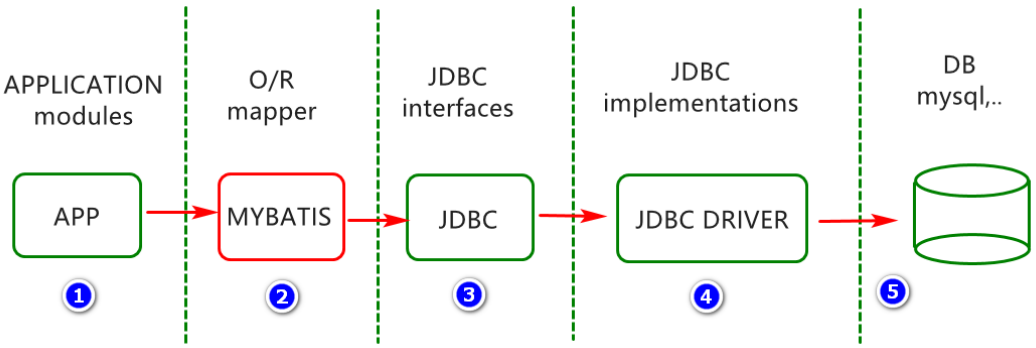
FRAMEWORK
第一章节
MYBATIS 原理剖析与实践

1.MyBatis 架构分析

1.1. 应用架构分析

Mybatis 是一个优秀的持久层框架，底层基于 JDBC 实现与数据库的交互，并在 JDBC 操作的基础上做了封装和优化，其应用架构如图所示：

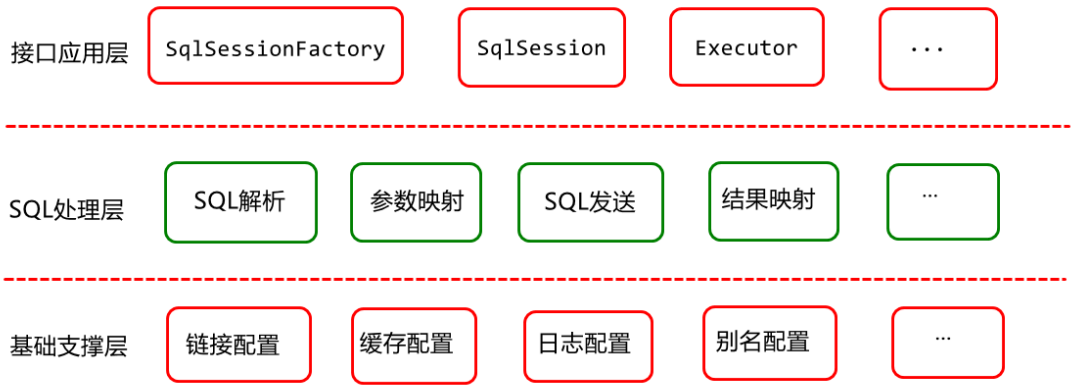


说明：

Mybatis 之所以能够成为互联网项目中持久层应用的翘楚，个人认为其核心竞争力应该是它灵活的 SQL 定制，参数及结果集的映射。

1.2. 产品架构分析

互联网项目中的任何一个框架都可以看成是一个产品，每个产品都有它自己的产品架构，Mybatis 也不例外，它的产品架构主要可以从接口应用，SQL 处理以及基础服务支撑等几个角度进行分析。如下图所示：

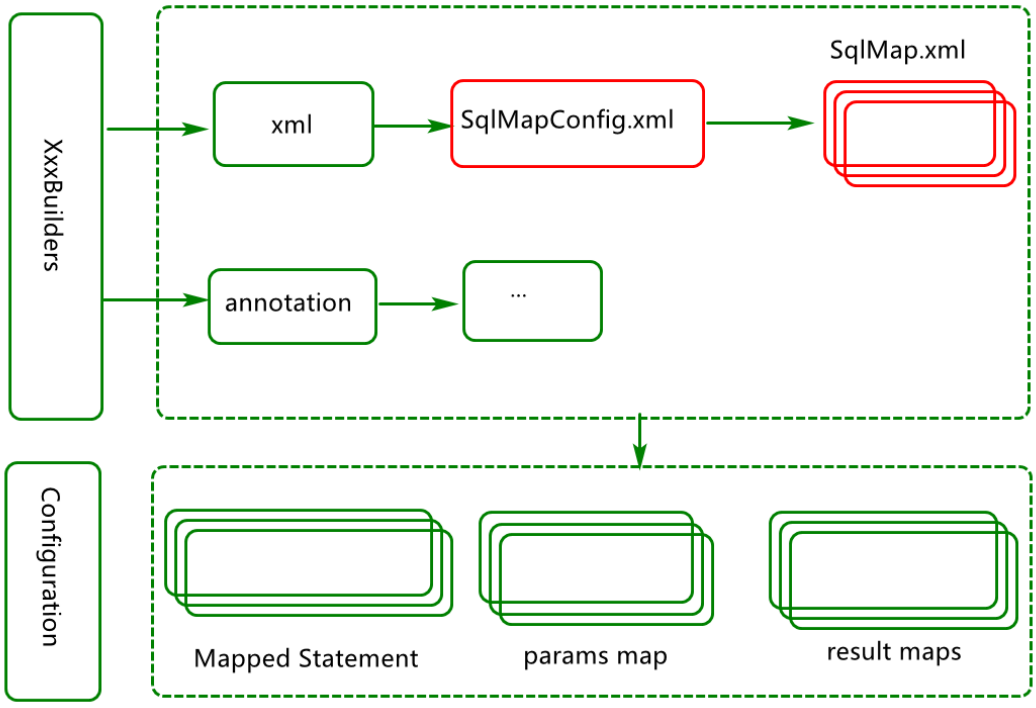


说明：

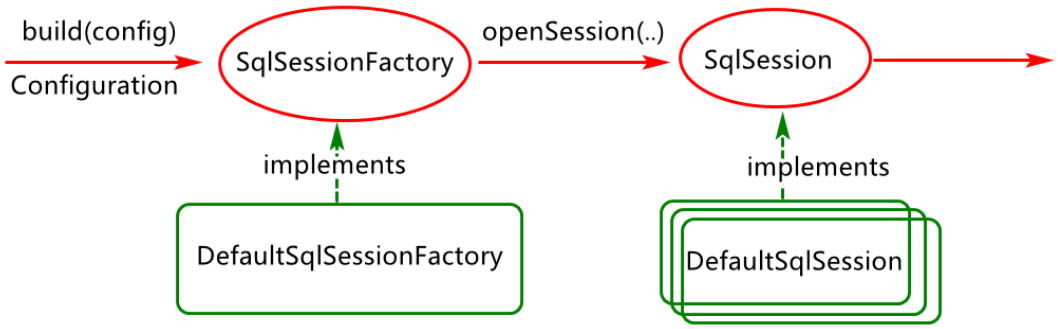
所有想成为平台架构师的程序员，在应用一个互联网框架的过程中都应对框架的设计理念，实现思路有一个很好的认知，并基于认知强化实践过程，拓展产品架构思维。

1.3. 技术架构分析

我们有了对产品架构的认知以后，还要了解产品架构背后的技术架构组成及原理，并能够迅速基于产品功能进行落地实现，然后优化和推广。



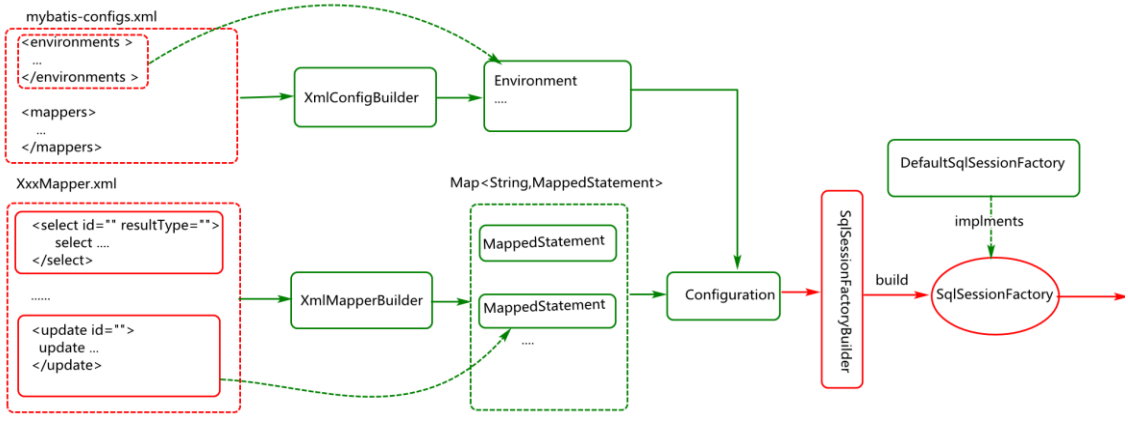
MyBatis 中的 API 架构



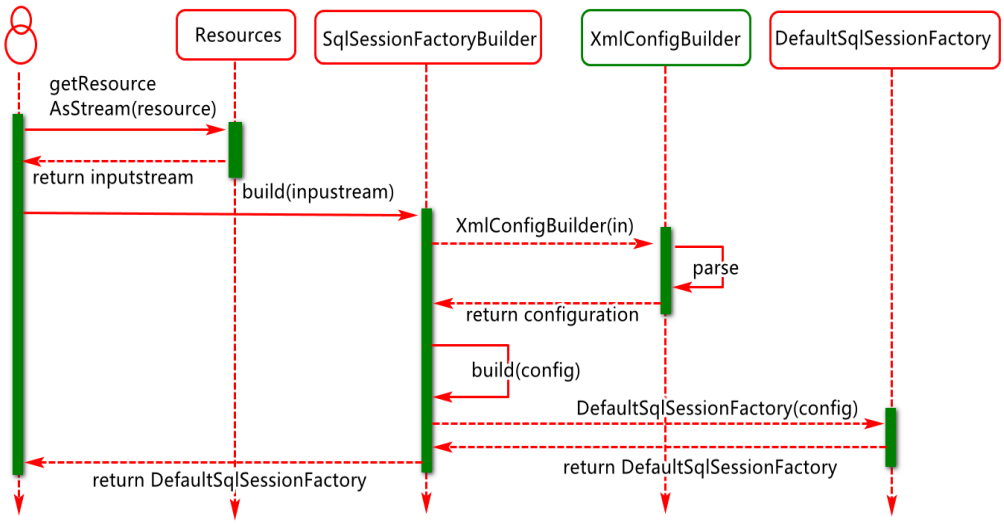
2.MyBatis 最佳实践

2.1. 核心对象应用

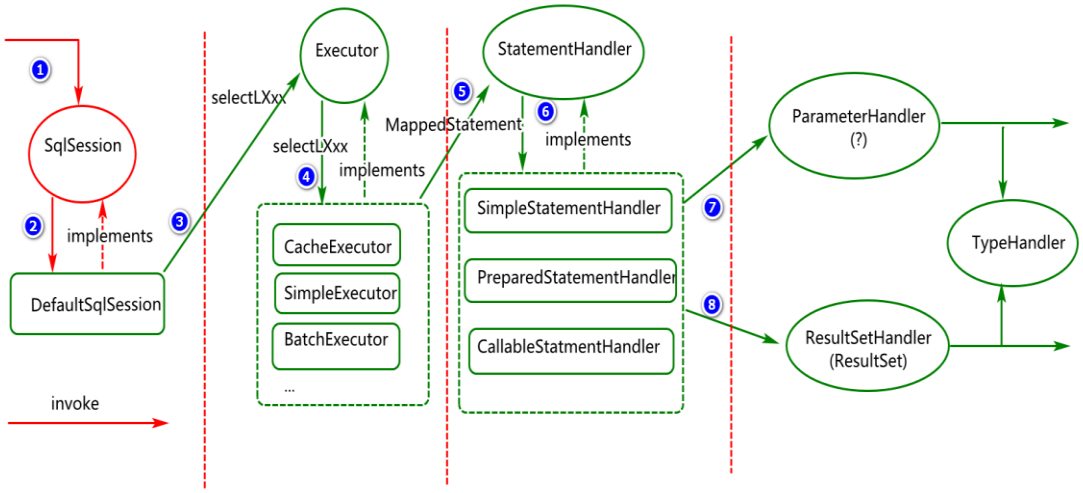
SqlSessionFactory 工厂对象创建分析



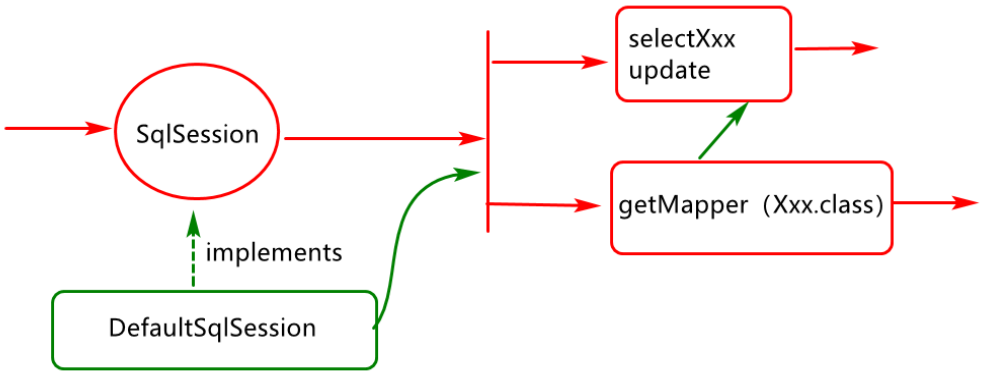
SqlSessionFactory 创建之时序图分析:



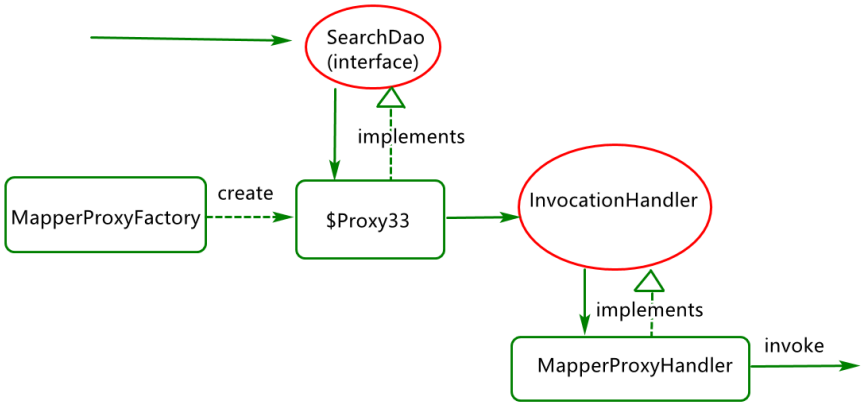
SqlSession 对象应用过程分析



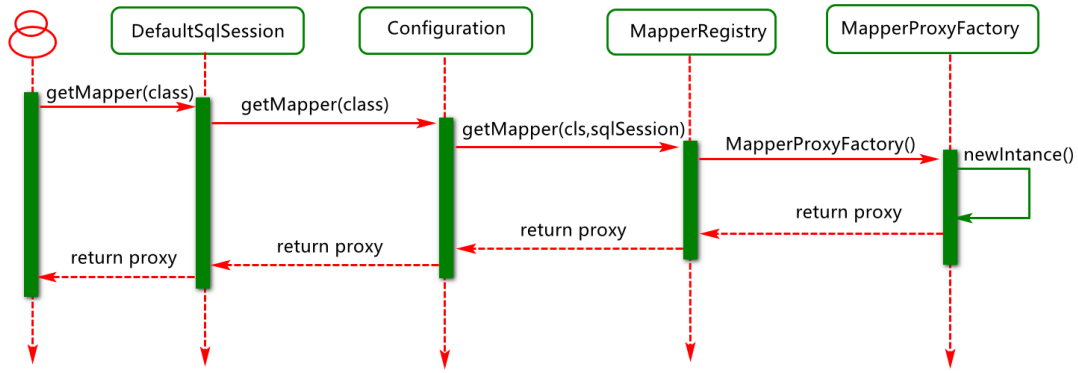
SqlSession 对象应用方式分析：



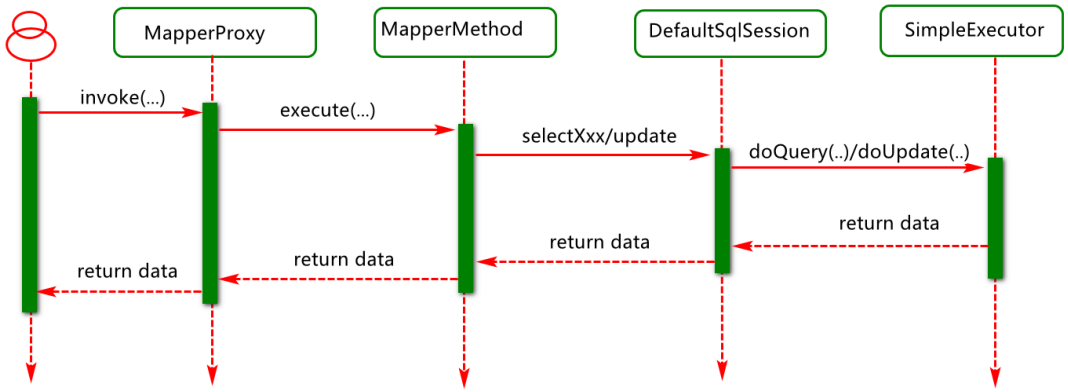
Mapper 接口代理对象创建分析



Mapper 代理创建之时序图分析



Mapper 代理对象数据访问应用过程分析



2.2. 其它对象应用分析

类型处理器(TypeHandler)对象应用:

类型转换器概述:

当我们借助 mybatis 将对象更新到数据库，或者从数据库获取数据，然后映射到内存中的对象时,中间都会涉及到 jdbc 数据类型与对象数据类型之间的转换,而这种转换在 MyBatis 中会借助 TypeHandler 类型处理器实现。

当然 MyBatis 中已内置了大部分的类型转换器，并在 TypeHandlerRegistry 类中进行了注册，但当我们业务中需要一些特殊的类型转换时，还是需要自定义类型转换器。

自定义拦截器的编写、配置及应用:
1) 实现 TypeHandler 接口或继承 BaseTypeHandler 类
2) 配置 TypeHandler 对象并应用

案例实现：实现文档管理系统中文档状态信息的更新与查询。

拦截器(Interceptor)对象应用:

Mybatis 拦截器概述:

mybatis 中提供一种插件应用机制，本质上就是采用责任链模式，通过动态代理使用多个拦截器，为目标业务添加一些扩展功能，例如对所有 dao 接口方法做一个日志记录和接口耗时记录。

拦截器对象编写，配置及创建：

1）实现拦截器接口 Interceptor，并对其要拦截的对象进行描述

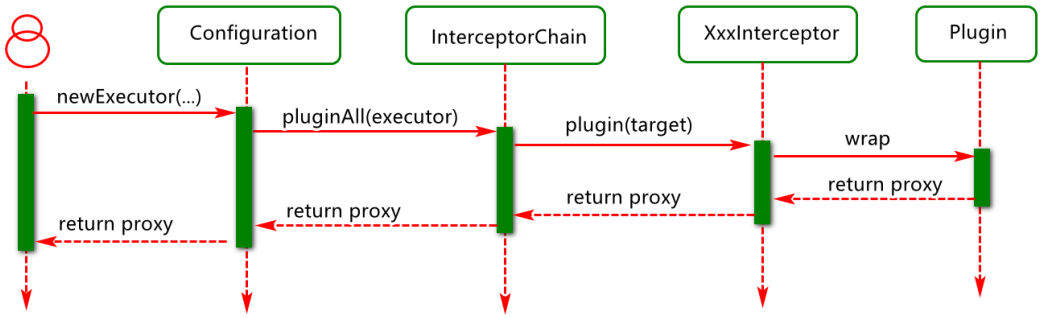
2）在 mybatis 配置文件中通过 plugin 标签进行配置

3）Mybatis 框架初始化时会创建拦截器对象然后添加到拦截器链

拦截器对象应用过程分析：

Mybatis 会在 Executor，StatementHandler，ResultSetHandler，ParameterHandler 对象创建时，假如有拦截器的定义会为指定对象创建代理对象，并在执行代理对象业务方法时，执行 Interceptor 对象的 intercept 方法。

例如：Executor 对象创建时的时序图如下：

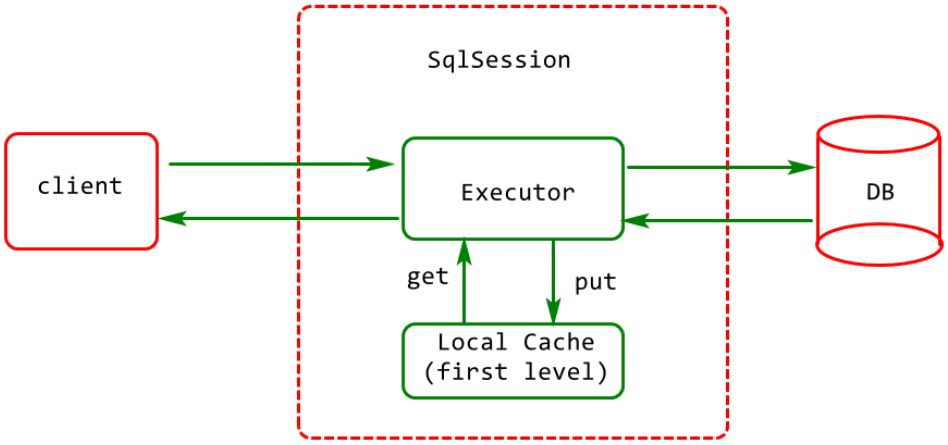


案例实现：基于拦截器实现分页插件，并掌握市场上 PageInterceptor 插件的应用原理。

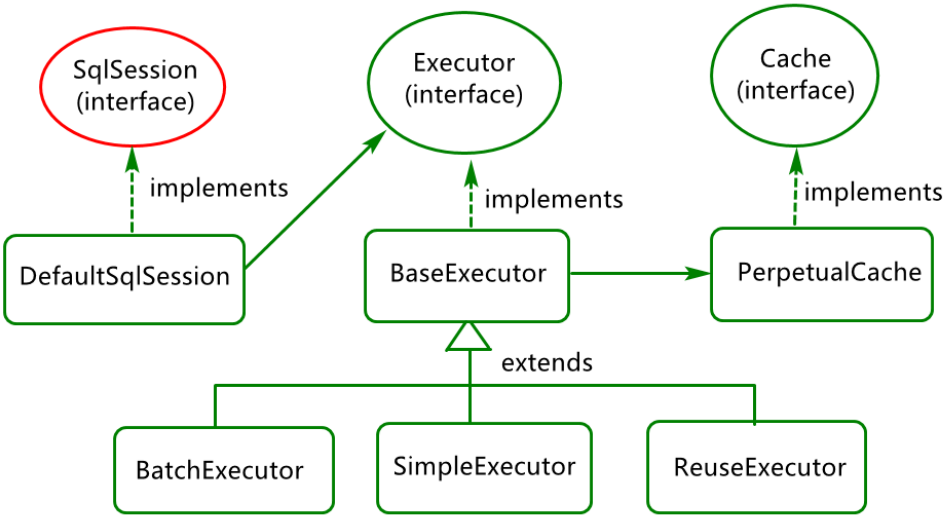
2.3. 缓存增强应用

2.3.1. 一级缓存

MyBatis 一级缓存应用基本架构：



MyBatis 一级缓存应用其应用类图结构如下：

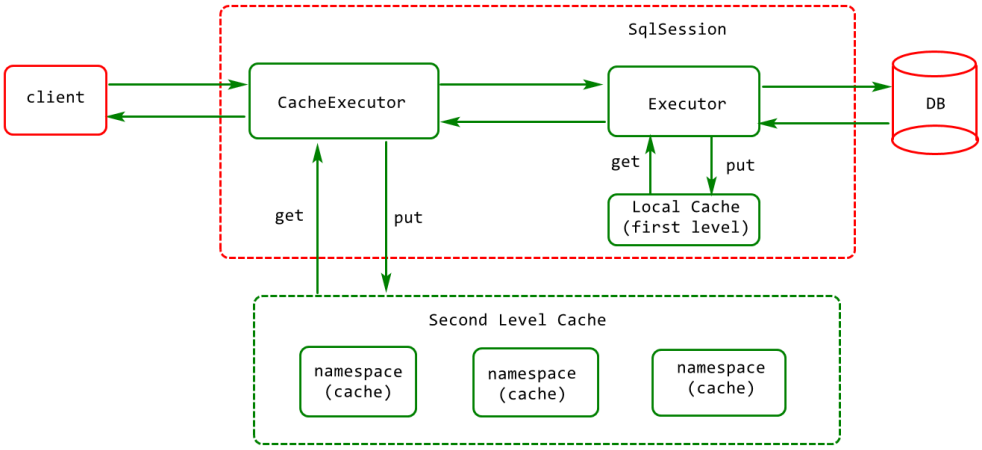


Mybatis 一级缓存应用特点：

1） 默认开启，其生命周期和 SqlSession 一致。
2） 内部设计是一个没有容量限定的 HashMap，在缓存的功能性上有所欠缺。
3） 最大范围是 SqlSession 内部，最小范围为 Statement.

2.3.2. 二级缓存

Mybatis 二级缓存应用基本架构：



MYBatis 二级缓存应用配置：

1） 在 MyBatis 的配置文件中开启二级缓存（默认开启）。

```
<setting name="cacheEnabled" value="true"/>
```

2） 在 MyBatis 的映射 XML 中配置 cache 或者 cache-ref 。

cache 标签用于声明这个 namespace 使用二级缓存，并且可以自定义配置。

```
<cache/>
```

a） type：cache 使用的类型，默认是 PerpetualCache，这在一级缓存中提到过。
b） eviction： 定义回收的策略，常见的有 FIFO，LRU。
c） flushInterval： 配置一定时间自动刷新缓存，单位是毫秒。
d） size： 最多缓存对象的个数。
e） readOnly： 是否只读，若配置可读写，则需要对应的实体类能够序列化。
f） blocking： 若缓存中找不到对应的 key，是否会一直 blocking，直到有对应的数据进入缓存。

cache-ref 代表引用别的命名空间的 Cache 配置，两个命名空间的操作使用的是同一个 Cache。

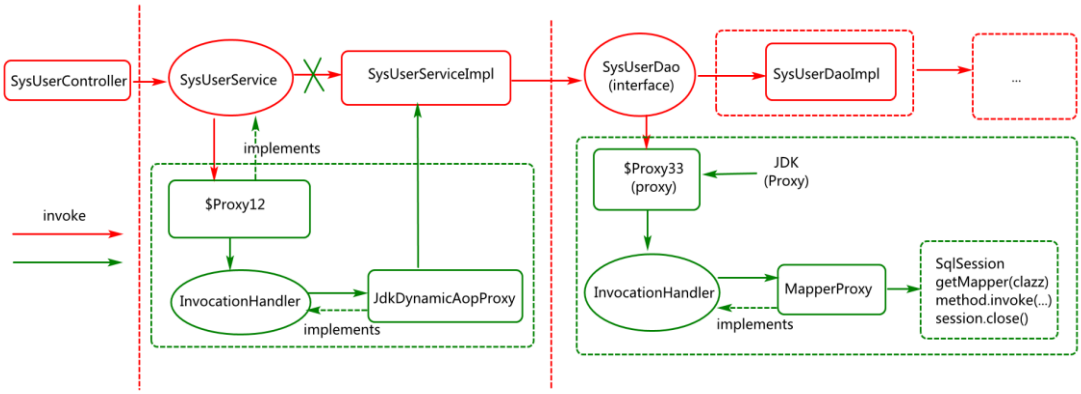
```
<cache-ref namespace="mapper.StudentMapper"/>
```

Mybatis 二级缓存应用特点：

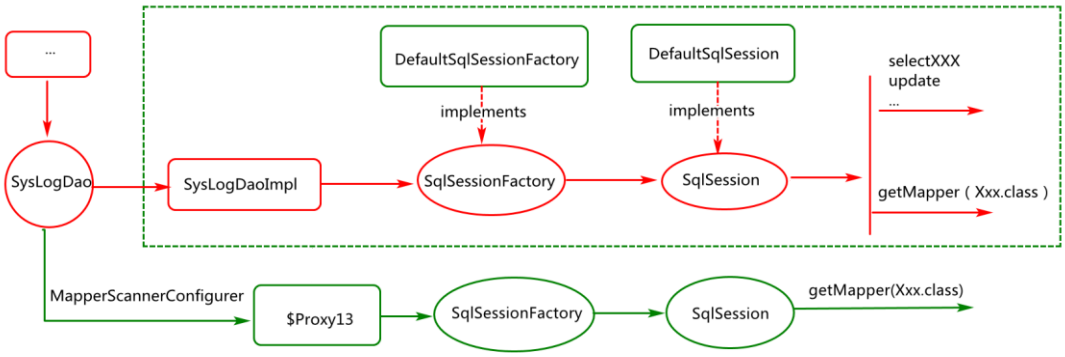
1) 二级缓存允许 SqlSession 之间缓存数据的共享，同时粒度更加的细，能够到 namespace 级别，通过 Cache 接口实现类不同的组合，对 Cache 的可控性也更强。
2) 在多表查询时，二级缓存可能会出现脏数据，有设计上的缺陷，安全使用二级缓存的条件比较苛刻。
3) 默认的 MyBatis Cache 实现都是基于本地的，分布式环境下会出现读取到脏数据，建议直接使用 Redis,Memcached 等分布式缓存可能成本更低，安全性也更高。

2.4. 项目整合应用

企业级项目中单体分层架构代码实现分析：



MyBatis 在项目整合中的基本架构应用分析，如下图所示：



基于对 mybatis 整合架构的理解，后续结合 spring 的学习，尝试使用 spring 整合 mybatis 做加强分析。

3.MyBatis 面试突破

3.1. 知识点应用面试分析

1．谈谈你对 mybatis 框架的理解？（可从 WWWH 角度展开）
2．说说 mybatis 持久化对象流程？（可从 save 操作说起）
3．说说 mybatis 中的核心 api 对象？（可从从会话工厂到会话过程说起）
4．谈谈你对 mybatis 中的动态 sql 的理解？（WWWH）
5．谈谈你对 mybatis 中#和$的异同点？（共性，特性）
6．Mybatis 中的批处理操作如何实现？（性能）
7．Mybatis 中可以自定义插件吗，如何实现？（扩展）
8．如何理解 mybatis 中的自定义类型转换器？（扩展）

3.2. 框架中设计模式应用

1．建造模式应用（Builder Pattern）

1) XmlConfigBuilder
2) XmlSatementBuilder

2．简单工厂应用

1) Configuration
2) DefaultObjectFactory

3．工厂方法应用

1) SqlSessionFactory
2) SqlSessionFactoryBean

3) DataSourceFactory
4) TransactionFactory

4. 单例模式应用

1) ErrorContext （线程内部单例）

5. 代理模式应用

1) session.getMapper(XxxDao.class);//JDK
2) 对象延迟加载时也会产生代理对象

6. 装饰模式

1) new CacheExcutor(excutor)

7. 适配器模式

1) Log 接口与日志框架应用

8. 责任链模式

1) Interceptor

9. 模版方法模式

1) BaseExecutor
2) SqlSessionTemplate

10. 策略模式

1) Cache （FifoCache,LruCache,…）

11. 组合模式

1) SqlNode