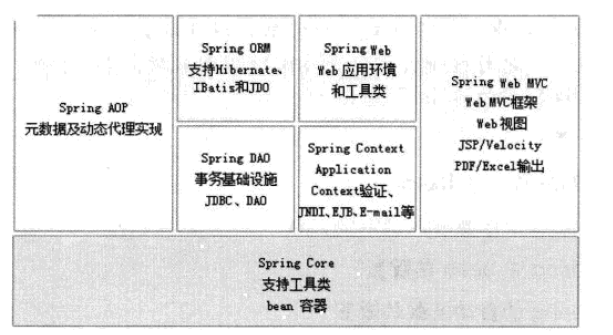
主要工程php开发，曾经看过一些，但因为很少上手去操作，基本也都忘了。

对spring 7大模块有概念性的理解，像spring mvc,dao,orm这些都是框架通行的逻辑。对于特色Ioc,AOP的原理有些理解就是了。



**核心容器(Spring core)**

核心容器提供spring框架的基本功能。Spring以bean的方式组织和管理Java应用中的各个组件及其关系。Spring使用BeanFactory来产生和管理Bean，它是工厂模式的实现。

Spring上下文(Spring context)

构建于Core封装包基础上的 Context封装包，提供了一种框架式的对象访问方法，有些象JNDI注册器。Context封装包的特性得自于Beans封装包，并添加了对国际化（I18N）的支持（例如资源绑定），事件传播，资源装载的方式和Context的透明创建，比如说通过Servlet容器。

**Spring面向切面编程(Spring AOP)**

通过配置管理特性，Spring AOP 模块直接将面向方面的编程功能集成到了 Spring框架中。所以，可以很容易地使 Spring框架管理的任何对象支持 AOP。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

**Spring DAO模块**

DAO模式主要目的是将持久层相关问题与一般的的业务规则和工作流隔离开来。Spring 中的DAO提供一致的方式访问数据库，不管采用何种持久化技术，Spring都提供一直的编程模型。Spring还对不同的持久层技术提供一致的DAO方式的异常层次结构。

**Spring ORM模块**

Spring 与所有的主要的ORM映射框架都集成的很好，包括hibernate、JDO实现、TopLink和IBatis SQL Map等。Spring为所有的这些框架提供了模板之类的辅助类，达成了一致的编程风格。

**Spring Web模块**

Web上下文模块建立在应用程序上下文模块之上，为基于Web的应用程序提供了上下文。Web层使用Web层框架，可选的，可以是Spring自己的MVC框架，或者提供的Web框架，如Struts、Webwork、tapestry和jsf。

**Spring MVC框架(Spring WebMVC)**

MVC框架是一个全功能的构建Web应用程序的MVC实现。通过策略接口，MVC框架变成为高度可配置的。Spring的MVC框架提供清晰的角色划分：控制器、验证器、命令对象、表单对象和模型对象、分发器、处理器映射和视图解析器。Spring支持多种视图技术

详解：

Ioc:

理解和介绍：

Inversion of control,控制反转是关于一个对象如何获取他所依赖的对象的引用，这个责任的反转

控制反转一般分为两种类型，依赖注入（Dependency Injection，简称DI）和依赖查找（Dependency Lookup）。依赖注入应用比较广泛。

ioc是目的，di是手段。ioc是指让生成类的方式由传统方式（new）反过来，既程序员不调用new,需要类的时候由框架注入（di）。

更多：see-17.2.2

AOP:面向切面编程

AOP把软件系统分为两个部分：核心关注点和横切关注点。

业务处理的主要流程是核心关注点，与之关系不大的部分是横切关注点。

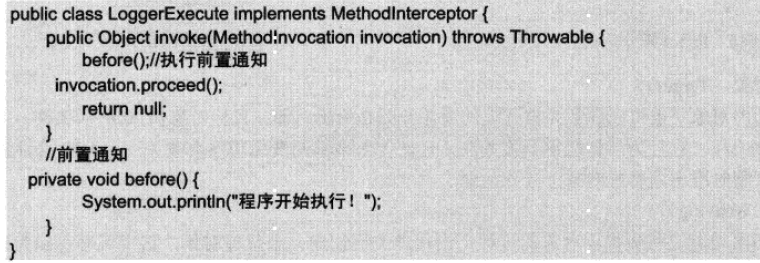
横切关注点的一个特点是，他们经常发生在核心关注点的多处，而各处都基本相似。比如权限认证、日志、事务处理。

原理：动态代理：jdk动态代理或cglib

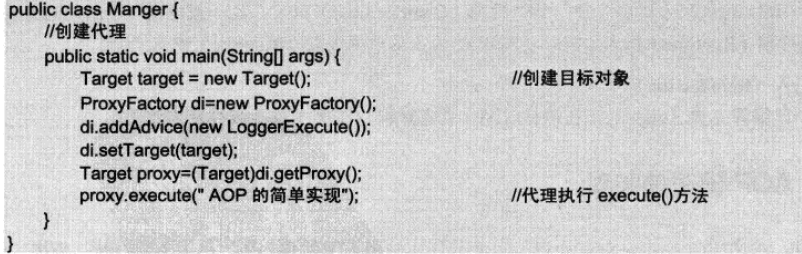
<http://blog.csdn.net/moreevan/article/details/11977115/>

简单实现：

通知：



调用



ProxyFactory本质是生成一个InvokeHandler，代理对象为Target,在invoke中执行method.invoke()时会相应执行loggerExecute的before等作为通知链，由此实现切入。

InvokeHandler的invoke主流程可以简述为：获取可以应用到此方法上的通知链（Interceptor Chain）,如果有,则应用通知,并执行joinpoint; 如果没有,则直接反射执行joinpoint。然后执行method.invoke(target, args)

作业帮aop，

Class xx extends advisor{

invoke(){

//实际逻辑

}

}

Advisor {

Execute(){ //通知，通过通知执行切面

//前置逻辑——如打点，用户信息，请求过滤

Invoke();

//后置逻辑

}

}

DAO/ORM

Spring mvc