SpringMVC与Servlet

1. SpringMVC就是Servlet呀。不过是一个Servlet调用不同的具体Bean来处理业务罢了。

2. springMVC的DispatcherServlet继承自 FrameworkServlet继承自HttpServletBean 继承自HttpServlet（也就是你说的servelt） 本质上是一样的东西

3.SpringMVC就是一个servlet处理所有的请求，在web.xml中的url-pattern是/或者/\*（拦截包括jsp）

4.拦截了所有的url，那么具体的url怎么处理呢？

答案就是，通过注解 @Controller 指定一个控制器类，业务逻辑的实现类（serviceImpl）通过@Service来标注，然后控制器类中声明好要注入的依赖对象的，通过@Autowired注解来动态地通过setter方法来注入对象（这样控制器和业务逻辑彼此都解耦合了）。 最后，控制器中转发业务请求的方法通过注解@RequestMapping来匹配具体的业务请求URL。

下面是简要示例：

Controller

@Controller

public class 项目名Controller {

@Autowired

Service1 service1; // 使用@Autowired标签必须规范命名 - 除首字母外和类名一致

@Autowired

Service2 service2;

@ResponseBody // 返回值以http response的正文形式返回给浏览器

@RequestMapping(value="/helloworld",method=RequestMethod.POST)

public String getHelloworld(@RequestParam(value="city") String city,

@RequestParam(value="project") String project) {

return service1.doSomething();

}

@ResponseBody

@RequestMapping(value="/hello",method=RequestMethod.GET)

public String getHello() {

return service2.doSomething();

}

}

Service

Service类负责具体业务, 最好是有个Service接口, 再实现这个接口. 面向接口编程的好处无需赘述.

public interface Service{

doSomething();

}

@Service

public class Service1 implements Service{

public String doSomething(){

// your business logic

}

}

5.