SpringMVC的核心流程：

FrameworkServlet.doGet()/doPost()/service()

FrameworkServlet.processRequest()

DispatcherServlet.doService()

DispatcherServlet.doDispatch()

DispatcherServlet.getHandler():

判断List<HandlerMapping> handlerMappings;是否为空

不为空则循环遍历该集合，拿到HandlerMapping hm;调用hm.getHandler();

这里因为Controller注册有不同的方式，不同的注册方式各自维持一个map集合，第一次遍历是为了拿到不同的map集合

getHandler()

找到与请求对应的Handler，最终返回给doDispatch()

DispatcherServlet.getHandlerAdapter();

判断handlerAdapters不为空后，遍历List<HandlerAdapter> handlerAdapters,拿到所有的adapter，调用adapter的supports方法，返回唯一一个使用的适配器

doDispatch拿到适配器后，调用adapter.handle，执行完Controller里的逻辑，返回ModelAndVeiw mv；

SpringMVC初始化

servlet 3.0规范，tomcat推荐使用配置类配置，取代了xml的配置文件

SpringMVC的Handler对象

扫描所有使用了@Controller注解的对象，并加载对象的方法，获取所有拥有@RequestMapping注解的方法，把方法和@RequestMapping里的value放入map当中去

SpringMVC会在Spring中通过类型去拿到所有在工厂中的适配器

通过向Spring容器中注册一个实现HandlerMapping接口的类就可以自定义适配器

Handler的注册方式两大类型，三种方式：

第一种类型，通过@Component注册到Spring工厂，拿到所有实现Controller接口，或。。。接口的类，以bean的name作为请求地址

第二种类型，通过@RequestMapping，注解实现注册，以注解的value作为请求地址

SpringMVC的Adapter

HttpRequestHandlerAdapter.supports();

该方法会判断Handler是实现了对应的接口；否则返回false；

Controller注册方式不止一种，所以根据不同的注册方式，要有不同的适配器与之对应