拦截器的作用很明确，好比玩游戏，你不可能直接体验游戏，你得登录，认证你是哪个玩家才能进入游戏，另一方面那些绝版道具不是谁都能拥有，你想要很简单，花钱，换句话，就是大家玩的一样的游戏，下的一样的资源包，你是土豪，你就能解锁更多新姿势，这些姿势那些平民就不能体验了。在我们web开发中，小的项目也有那么几十个页面，大的项目成百上千个页面也不是没有，同样的，最基本的我们也要有用户的登录过程，有些页面你总得登录才能访问，更有要涉及到一些敏感信息，只给部分有特权的用户才能访问的页面，我们总不能任谁在地址栏一输入那个页面就进去了吧 ？？？ 这里就是我们拦截器大显身手的地方了。呵呵，你不登录就想访问我们系统的其他页面？？？（手动黑人问号），滚回去登录，你没有访问我们特权人士才能访问的页面的特权就想来？？？ 小样，冲够了钱再来吧（黎子：--，这小哥游戏玩疯了吧，作者：qnmlgb）。

嗯，就是这么回事了（黎子：哪回事啊，这就这回事了？？？我还一脸懵逼啊）。不要着急，上面就是废话（毫不脸红的说）（黎子黑线中）。我们springmvc的拦截器类是基于HandlerInterceptor接口的，一共实现了三个方法：

```
@Override
public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler, Exce
        throws Exception {


@Override
public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler, ModelAndVie
        throws Exception {


@Override
public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Ex
```

三个方法执行的顺序是这样的，Controller类处理之前会先调用preHandle，该方法返回true时会继续进行即找相应的Controller类去，为false时自然就是拦截成功了，乖乖按提示去操作吧。处理完Controller类后就会调用postHandle，即在DispatcherServlet进行视图返回渲染前调用，可以对ModelAndView操作。最后调用afterCompletion，进行资源清理。这里我们只需要先记住三个方法执行顺序以及执行的地方，主要来聊聊preHandle的体会（黎子：感情你就学了个preHandle就在这装大神了？？？作者：你滚！！！）。

拦截拦截，顾名思义，我们肯定要获取地址，然后对地址先判断，再判断权限，最后给予是允许还是拒绝的信号。

先判断地址，不是所有地址都需要拦截,我们就需要获取当前请求地址和设为不拦截的请求比对，

```
//不拦截loginForm login
private static final String[] IGNORE_URI={"/loginForm","/login"};
```

```
//获取请求路径进行判断
String servletPath =request.getServletPath();
System.out.println(servletPath);
//判断请求是否需要拦截
for(String s:IGNORE_URI){
    if(servletPath.contains(s)){
        flag=true;
        break;
    }
}
```

当前请求如不需要拦截当然直接放行。

再来判断权限，

```
//拦截
if(!flag){
    //获得session中的用户
    User user=(User)request.getSession().getAttribute("user");
    //判断用户是否已经登录
    if(user==null){
        System.out.println("拦截开始...");
        request.setAttribute("message", "请先登录再访问网站");
        request.getRequestDispatcher("loginForm").forward(request, response);
    }else{
        System.out.println("放行...");
        flag=true;
    }
}
```

这里举个黎子，哦不例子（黎子：你找打）是判断用户是否登录，也是用的最多的地方，首先获取session中的“user”，若没有，那便为没有登录，没登录就会进行loginForm请求，注意，这里是直接进行loginForm请求，相当于当前请求进行到一半停住了，请看下面的截图，

```
preHandle 执行了...
/main
拦截开始...
preHandle 执行了...
/loginForm
true
loginFrom方法执行了...
postHandle 执行了...
afterCompletion 执行了...
false
```

拦截到main请求并进行到“拦截开始”，并没有结束，而是直接又去拦截loginForm请求，并且返回了true，同时依次执行了postHandle，afterCompletion，最后再回到main请求的拦截，返回了false，结束了越权进行了main请求，我们的页面是这样的。

登陆界面

请先登录再访问网站

登录名：

密码：

登录

哦，差点忘了，我们拦截器写完了，还要在springmvc的配置文件中配置一下，这就是我们springmvc拦截器的另一个魅力所在，即插即拔，

```
<mvc:interceptors>
    <mvc:interceptor>
        <mvc:mapping path="/*" />
        <!-- 使用bean定义一个Interceptor，直接定义在mvc:interceptors根下面的Interceptor将拦截所有的请求 -->
        <bean class="com.jlb.interceptor.AuthorizationInterceptor" />
    </mvc:interceptor>
</mvc:interceptors>
```

想要什么样的拦截，就在这里添加一下即可，不想要在这里删掉就行，是不是很方便。

嗯，这次真的大概就是这样了（黎子：真是烂到家了，不过念在你也是新手，原谅你了 作者：多谢黎子的谅解，第一次写，文笔，理解方面，都还有很多欠缺，不过嘛，目前也只面向私人学习，练手嘛）。

最后附上项目

SpringMVCDemoIn…r.zip
10.14MB