URL:

https://portal.aia.com.sg/wps/wcm/connect/f2031a80445a4cd28c54cfa547e8e9e9/Application+and+Product+Summary+Booklet+for+Glow+of+Life+(Feb+2017).pdf?MOD=AJPERES

https://sindcslfwm01.aia.biz/wps/wcm/connect/212e0b80447a5051accead82c7b210aa/PD\_PT+0022104\_Conf+FSC-IR+Rpt.indd.pdf?MOD=AJPERES&CACHEID=212e0b80447a5051accead82c7b210aa

# URL权限控制

## 开发思路

在系统开发过程中需要考虑的一个重要的问题就是权限问题，权限问题也是安全问题的一个范畴，我们要求在用户登录系统之后，要控制用户可以访问的系统资源，使得用户只可以访问到系统事先分配好的资源；这里的资源可以是一个URL地址，也可以是页面上的菜单和按钮等。对于实现权限的控制有多种方案，说明一下通过URL拦截的方式进行权限控制的实现方案。

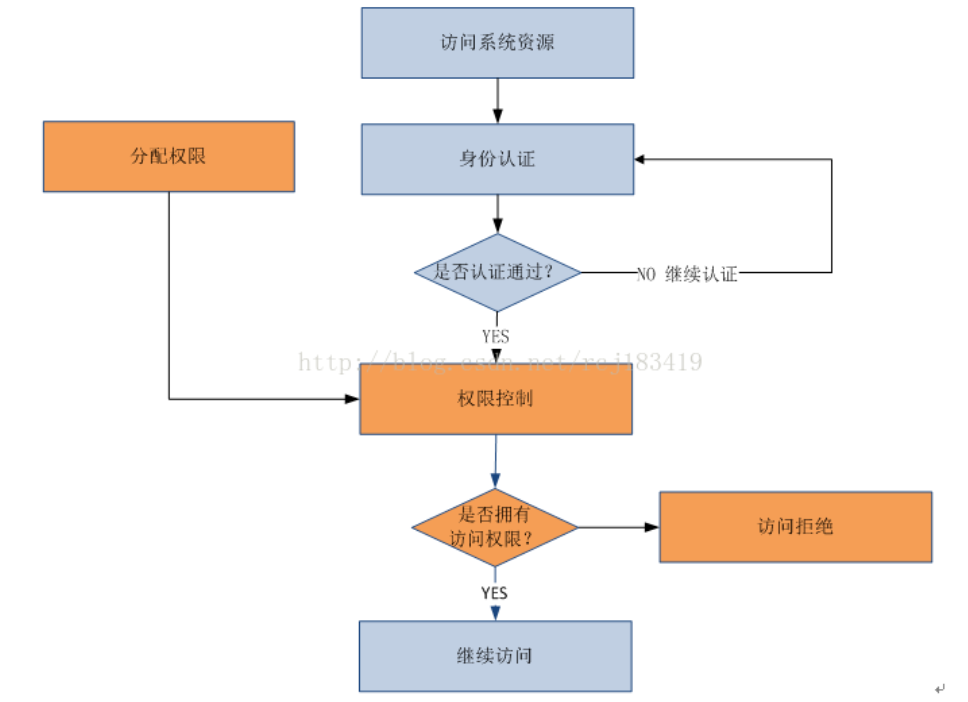
## 基本流程

对于权限的控制可以分为两个步骤：认证和授权

认证：即用户登录系统的时候对用户的身份信息进行判别。

授权：即在用户登录成功后为用户分配可以访问的资源。

流程图：根据用户的认证和授权过程抽象出如下流程图



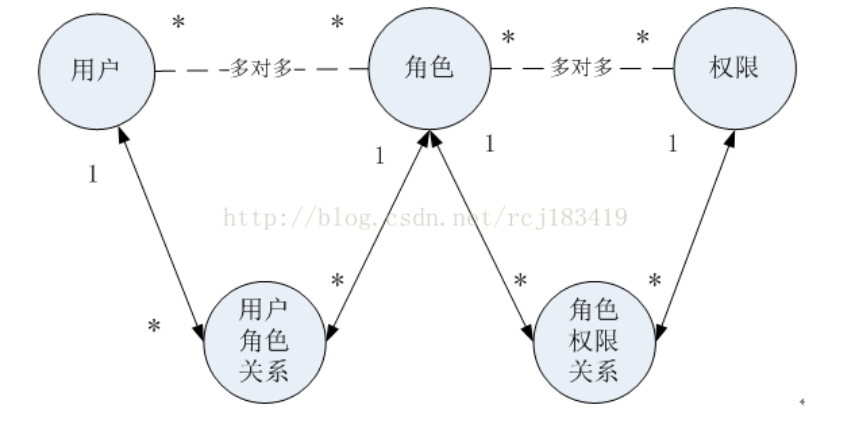
## 通用模型

根据我们对权限控制的要求，我们可以抽取出如下数据模型：

主体：用户，程序等，包括账号和密码等属性。

资源：URL，菜单，按钮等。

角色：为了方便资源和主体之间的关系管理，我们一般会在它们之间抽取出一个角色实体，一个角色就是一类主体，通过角色可以实现对主体的分组管理，这样可以更加方便的对主体和其所对应的资源进行管理（扩张和修改）。模型结构如下图所示：



说明：用户和角色是多对多的关系，角色和权限是多对多关系。

## 实现过程

### 1、定义用户身份和基本操作

这里我们创建一个用户的身份实体ActiveUser，用来存放用户的身份信息，在用户登录成功后将该身份信息存放到session当中。



### 2、公开访问地址配置

### 3、公共访问地址配置

### 4、认证拦截器

通过认证拦截器对用户身份信息进行判断。

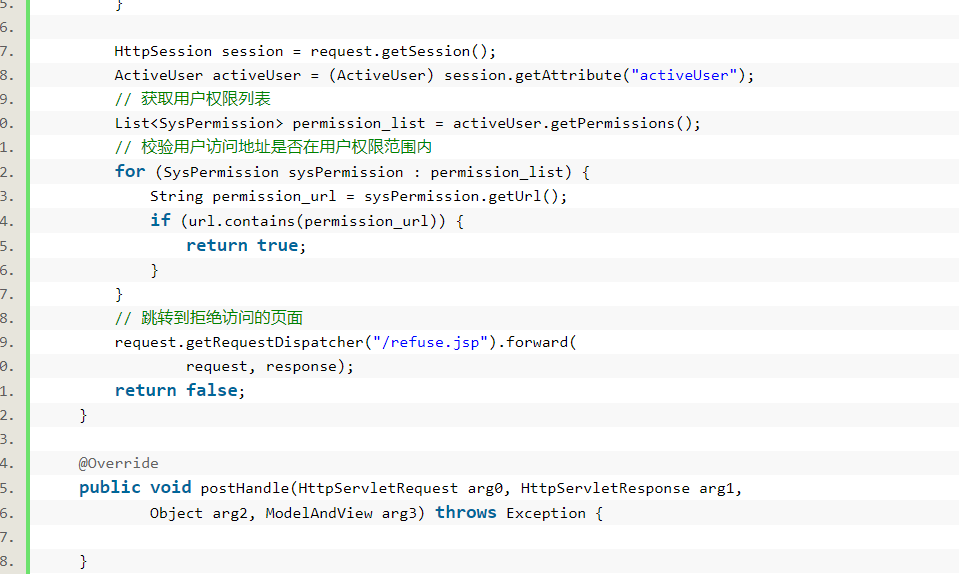


说明：判断session中是否包含用户成功登录的信息。

### 5、授权拦截器

通过授权拦截器，判断用户是否具有访问资源的权限。





### 6、配置拦截器

将拦截器配置起来，使其工作。



## 总结

这里主要是通过两个拦截器实现了认证和授权，其优点是可以不必依赖于框架实现。对于拦截器我们也可以通过web提供的filter实现；缺点在于对于系统配置很多的URL，或者在系统初始化时将URL设置到数据库中，再有就是对于访问地址的变动要同时改变配置，维护相对不易。

## 第二种方法

通过startsWith与endsWith方法可以在自定义filter中排除过滤掉静态文件和一些不用验证的URL。

代码如下：

Web.xml文件配置



Java配置

Jboss-seam.jar