**nodejs+express框架实现登录过滤器以及用户登入状态的保存**

之前用[express](https://so.csdn.net/so/search?q=express&spm=1001.2101.3001.7020)框架搭建一个后台管理应用，主要是学习node如何编写接口。今天带大家学习node如何实现过滤器，以及用户信息状态如何存储。  
先来了解中间件：cookie-parser模块与express-[session](https://so.csdn.net/so/search?q=session&spm=1001.2101.3001.7020)模块。  
cookie-parser模块：这个是为了方便操作客户端中的cookie值，区别于session。

npm i cookie-parser --save

app.js文件

引入中间件

var cookieParser = require('cookie-parser')

使用中间件

app.use(cookieParser('123456'));

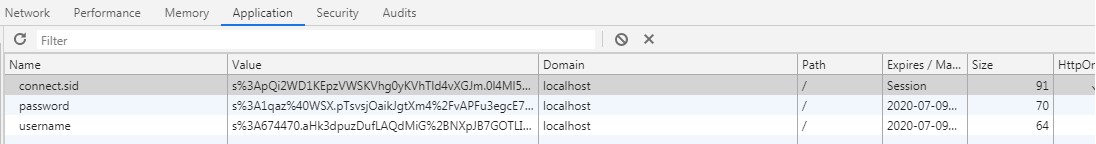
设置cookie

res.cookies('key','value',option);具体的option配置可查看其它博文

是否传参数的影响：

1、在使用中间的时候是否传参其实就是是否加密，那么传参之后的加密效果如下图。

2、在设置cookie值时，如果将属性signed设置为true时，则必须传参。

在后端种好cookie之后，浏览器会出现如下图片：  
在这里插入图片描述  
express-session模块：session 是另一种记录客户状态的机制， 不同的是 Cookie 保存在客户端浏览器中， 而 session 保存在服务器上。

引入express-session

npm i express-session --save

app.js文件

var session = require("express-session");

app.use(session({

secret: '123456',

resave: false,

saveUninitialized: true

}));

以上代码就将两个中间件引入当自己的项目当中了，那如何使用，并且在哪里使用。如下例子可以说明：  
注册接口：将用户输入的信息存到数据库当中。

var express = require('express');

const request = require('request');

const queryString=require('query-string');

var router = express.Router();

var db=require('../sql/db.js');

router.get('/',(req,res,next)=>{

let username,password;

if(Object.keys(req.query).length===2){

username=req.query.name;

password=req.query.password;

db({database:'hhl'},"INSERT INTO user(username,password) VALUES ('"+username+"','"+password+"')",function(result,fields){

res.send('注册成功！！')

})

}

})

module.exports=router;

登入接口代码：在这个接口当中，查询数据库，并且将查询的结果种到cookie当中，存在session中，那么下次想获取用户信息则直接在session中获取。

login.js

var express = require('express');

var router = express.Router();

var db=require('../sql/db.js');

router.get('/', function(req, res, next) {

let name,password;

if(Object.keys(req.query).length==2){

name=req.query.name; //用户名

password=req.query.password; //用户密码

db({database:'hhl'},"SELECT username,password FROM user WHERE username='"+name+"' AND password='"+password+"'",function(result,fields){

if(!result.length){

res.send('登入失败！！');

}else{

req.session.auth\_username=name;

req.session.auth\_password=password;

res.cookie('username',name, {maxAge:1000 \* 60 \* 60 \* 24 \* 7,signed:true}); //设置cookie的保存时间，7天为例

res.cookie('password',password, {maxAge: 1000 \* 60 \* 60 \* 24 \* 7,signed:true}); //console.log(req.signedCookies);

res.send('登入成功！！');

}

})

}

});

module.exports = router;

设置完成之后的cookie值如图片  
在这里插入图片描述  
注销接口：  
将session，cookie的值都删除。

var express = require('express');

const queryString=require('query-string');

var router = express.Router();

var db=require('../sql/db.js');

router.get('/',(req,res,next)=>{

let user=req.session.auth\_username;

let pass=req.session.auth\_password

res.cookie('username',user, {maxAge:0,signed:true});

res.cookie('password',pass, {maxAge:0,signed:true});

req.session.auth\_password=undefined;

req.session.auth\_username=undefined;

let {username,password}=req.signedCookies;

console.log(req.session.auth\_username && req.session.auth\_password,username,password)

res.send('注销成功！！');

})

module.exports=router;

获取图片接口：（这个接口就是用来做测试的）

var express = require('express');

var router = express.Router();

var path = require('path');

var multiparty = require('multiparty');

router.get('/',(req,res,next)=>{

res.send({

status:"200",

msg:"成功！",

img:'http://localhost:3000/images/home1.png'

});

})

那么重点关注对象----过滤器怎么实现。代码如下  
session关闭浏览器这信息就没了，如果使用cookie来实现信息的有效保存。  
实现思路：  
1、判断是否存在session  
是 next  
否 判断是否有cookie  
是 将cookie值取出来赋值给session 并且next  
否 获取请求地址，白名单login，relogin接口 next

app.use(function(req,res,next){

if(!(req.session.auth\_username && req.session.auth\_password)){

console.log('无session');

if(req.signedCookies.username && req.signedCookies.password){

console.log('有cookie')

let {username,password}=req.signedCookies;

req.session.auth\_username=username;

req.session.auth\_password=password; //将cookie的值存在session里

next();

}else{

let arr=req.url.split('/');

console.log('无cookie',arr);

let index=arr && arr.findIndex((item)=>{

return (item.indexOf('login')!=-1 || item.indexOf('relogin')!=-1);

});

if(index!==-1){

next();

}else{

return res.status(401).json({

msg: '没有登入，请先登入'

})

}

}

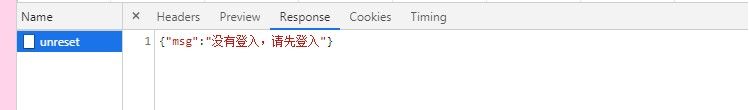
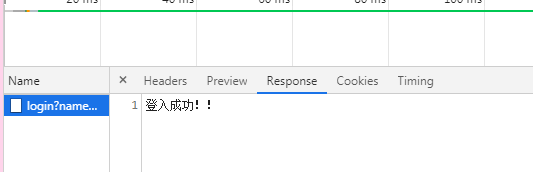
}else{

console.log('有session',req.session.auth\_username && req.session.auth\_password);

next();

}

})

代码思路检验：  
1、还没登入点注销出现如下图：说明被拦截  
  
2、点击登入的图片：  
  
3、点击登入之后，获取图片：  
在这里插入图片描述  
4、点击注销：  
  
5、最后再获取图片一下：



过滤器以及登入注销的实现就完成啦~~来总结一下：  
坑1：过滤器一定要放在路由导向之前。  
坑2：sql拼接一定要心细心细再心细。  
优化建议：  
如上的代码就是将具体功能实现了而已，并不完美。可以将过滤再起一个文件夹，因为项目当中可以用的的过滤器，拦截器不止一个。最后在app.js引入就行，这样看的舒服。再者，每个接口的都需要优化，比如注册接口的用户账户判重，密码强弱设置，以及友好的用户提示等等之类的功能。

专栏目录

[movie-site：基于Nodejs + Express + MongoDB + jQuery + Bootstrap构建的电影网站！](https://download.csdn.net/download/weixin_42138376/15033390" \t "_blank)

[本文主要介绍NodeJS+Express+Mysql 实现POST和GET请求的增删改查](https://download.csdn.net/download/qq_38209578/10667760" \t "_blank)

[nodejs使用express模块使用过滤器添加跨越访问控制](https://blog.csdn.net/weixin_30539625/article/details/97418550" \t "_blank)

[Node.js 教程第八篇——Express 过滤器](https://blog.csdn.net/weixin_34128237/article/details/88858934" \t "_blank)

[基于Express框架的博客系统，适用于初学者学习，参考。](https://download.csdn.net/download/weixin_45815467/12111467" \t "_blank)

[基于Express框架的后台管理系统](https://download.csdn.net/download/weixin_44224921/12874606" \t "_blank)

[协同过滤算法的个性化智能图书推荐系统(Express+源码+调试)-CSDN博...](https://blog.csdn.net/vcd604/article/details/137666943" \t "_blank)

[基于node.js的express使用mysql语句在插入数据时防重插入](https://blog.csdn.net/qq135595696/article/details/120621234" \t "_blank)

[nodejs使用express-session做登录状态保持](https://blog.csdn.net/weixin_44025657/article/details/126558575" \t "_blank)

[编写一个POST请求非法参数过滤中间件(express框架下)\_js非法请求-C...](https://blog.csdn.net/weixin_44176696/article/details/108006731" \t "_blank)

[Node.js express框架中间件, 应用级中间件(路由过滤),内置中间件(静态...](https://blog.csdn.net/houyanhua1/article/details/79469397" \t "_blank)

[nodejs的express使用全局拦截器保存用户请求和响应日志](https://blog.csdn.net/unhejing/article/details/124152232" \t "_blank)

[nodejs的express框架中如何对用户密码进行加密存储（md5加密存储用户信息）](https://blog.csdn.net/weixin_45745641/article/details/127469593" \t "_blank)

[nodeJS+Express+mysql\_node.js - 按字母顺序对数据进行排序并使用 exp...](https://blog.csdn.net/rolinabc/article/details/107003549" \t "_blank)

[node配置express服务器、拦截路由、连接数据库,返回前端数据全过程](https://blog.csdn.net/WAIXINGHAIKE/article/details/121846535" \t "_blank)

[使用express-session来保存登录状态](https://blog.csdn.net/qq_43414697/article/details/83755626" \t "_blank)

[nodejs express 登录中间件 中间件的原理及实现](https://blog.csdn.net/kingrome2017/article/details/77895820" \t "_blank)

[node和express实现mySql模糊搜索\_node+express+mysql实现模糊查询-CSDN...](https://blog.csdn.net/Android_boom/article/details/126284523" \t "_blank)

[NodeJS+Express+Mysql实现用户登录和注册。](https://download.csdn.net/download/qq_38209578/10662752" \t "_blank)

[nodejs+Express+mssql](https://download.csdn.net/download/li_yibin/20087373" \t "_blank)

07-08

[nodejs+Express+mssql,SQL Server实现增删改查](https://download.csdn.net/download/li_yibin/20087373" \t "_blank)

**热门文章**

* [开箱即用：Vue和Express搭建的一个功能完善的前后端分离项目 [https://csdnimg.cn/release/blogv2/dist/pc/img/readCountWhite.png](https://blog.csdn.net/Miss_hhl/article/details/107531530) 3450](https://blog.csdn.net/Miss_hhl/article/details/107531530)
* [手把手教你通过express框架搭建后台+vue框架搭建前台 [https://csdnimg.cn/release/blogv2/dist/pc/img/readCountWhite.png](https://blog.csdn.net/Miss_hhl/article/details/106825918) 3202](https://blog.csdn.net/Miss_hhl/article/details/106825918)
* [vue@cli3如何配置来区别生产环境，开发环境，测试环境 [https://csdnimg.cn/release/blogv2/dist/pc/img/readCountWhite.png](https://blog.csdn.net/Miss_hhl/article/details/103584825) 2050](https://blog.csdn.net/Miss_hhl/article/details/103584825)
* [nodejs+express框架实现登录过滤器以及用户登入状态的保存 [https://csdnimg.cn/release/blogv2/dist/pc/img/readCountWhite.png](https://blog.csdn.net/Miss_hhl/article/details/106946911) 1756](https://blog.csdn.net/Miss_hhl/article/details/106946911)
* [webpack4搭建vue项目引入element-ui库，引入样式报错问题解决。 [https://csdnimg.cn/release/blogv2/dist/pc/img/readCountWhite.png](https://blog.csdn.net/Miss_hhl/article/details/103768314) 1683](https://blog.csdn.net/Miss_hhl/article/details/103768314)