# 一、复习

cookie是在res中设置，req中读取的。第一次的访问没有cookie。

cookie的存储大小有限，kv对儿。对用户可见，用户可以禁用、清除Cookie、可以被篡改。

cookie用来制作记录用户的一些信息，必须购买历史、猜你喜欢。

HTTP是无状态的协议，所以两次的访问，服务器不能认识到是同一个客户端的访问，就要用cookie来巧妙的解决这个问题。

Session就是利用cookie，实现的“会话”。就是第一次访问的时候，可以在服务器上为这个用户缓存一些信息，别的用户是不能看见这个用户的信息的。服务器会下发一个秘钥（cookie），客户端每次访问都携带这个秘钥，那么服务器如果发现这个秘钥吻合，就能够显示这个用户曾经保存的信息。

登陆就是用Session来制作的。任何语言的session都是透明的，不会体现cookie机理。

|  |
| --- |
| 1. var session = reqiure("express-session"); 2. app.use(session({ 3. ..一些配置 4. ..一些配置 5. ..一些配置 6. })); 7. app.get("/",function(req,res){ 8. console.log(**req**.sission.login); 9. }); 10. app.get("/login",function(req,res){ 11. **req**.session.login = "1"; 12. }); |

都是使用req对象。

# 二、加密

**永远不要用明码写密码**。CSDN今年泄露用户密码了，并且泄露的明码。

黑客拿到的用户的密码的加密信息，所以也没用。因为他无法翻译成为明码。

MD5加密是函数型加密。就是每次加密的结果一定相同，没有随机位。

特点：

● 不管加密的文字，多长多短，永远都是32位英语字母、数字混合。

● 哪怕只改一个字，密文都会大变。

● MD5没有反函数破解的可能，网上的破解工具，都是通过字典的模式，通过大量列出明-密对应的字典，找到明码。两次加密网上也有对应的字典。所以我们不要直接用一层md5，这样对黑客来说和明码是一样。

MD5常用于作为版本校验。可以比对两个软件、文件是否完全一致。

node中自带了一个模块，叫做crypto模块，负责加密。

首先创建hash，然后update和digest：

|  |
| --- |
| 1. var md5 = crypto.createHash('md5'); 2. var password = md5.update(fields.password).digest('base64'); |
|  |

# 二、图片处理

<http://www.graphicsmagick.org/>

GraphicsMagick is the swiss army knife of image processing. 瑞士军刀。

只要服务器需要处理图片，那么这个服务器就要安装graphicsmagick软件。免费的。

装完之后，可视化工具一点用都没有，从桌面上删除。我们要把安装目录设置为环境变量。

控制台CMD命令：

|  |
| --- |
| 1. //格式转换 2. gm convert a.bmp a.jpg |

|  |
| --- |
| 1. //更改当前目录下\*.jpg的尺寸大小，并保存于目录.thumb里面 2. gm mogrify -resize 320x200 danny.jpg |

nodejs要使用graphicsmagick，需要npm装一个gm的包。

node.js命令

node.js缩略图的制作：

|  |
| --- |
| 1. var fs = require('fs'); 2. var gm = require('gm'); 3. gm('./danny.jpg') 4. .resize(50, 50,"!") 5. .write('./danny2.jpg', function (err) { 6. if (err) { 7. console.log(err); 8. } 9. }); |

node.js头像裁切：

|  |
| --- |
| 1. gm("./danny.jpg")**.crop(141,96,152,181)**.write("./2.jpg",function(err){ 2. **//141 96 是宽高 。 152 181是坐标** 3. }); |