# 一 请求

## 0 表单的enctype

application/x-www-form-urlencoded 默认提交方式，即url编码方式

text/plain 用的很少，纯文字提交

multipart/form-data 提交文件

## 1 GET请求

ES6支持解构赋值，获取url与参数的方式更加简单：

const http = require('http');  
const url = require('url');  
  
http.createServer((req,res)=>{

*//true代表直接将结果解析为json*  
 let {pathname,query} = url.parse(req.url,true);  
 console.log(pathname); *//输出请求地址 /get* console.log(query); *//参数对象 {name:'lisi'}* res.end('hello');  
}).listen(8000);

## 2 POST请求

Node为了让整个过程非阻塞，将POST数据拆分成很多小块，然后通过触发特定的事件，将小块数据有序传递给回调函数，是用on事件来监听。

前台代码：

<form action="http://localhost:8000" method="post">  
 <input name="username" type="text">  
 <input type="submit" value="提交">  
</form>

const http = require('http');  
const querystring = require('querystring');  
http.createServer((req,res)=>{  
 let str = '';  
 req.on('data',data=>{  
 str += data;  
 });  
 req.on('end',()=>{  
 let postData = querystring.parse(str);console.log(postData); *// 输出json格式* });  
 res.end();  
}).listen(8000);

注意：通常我们需要先判断客户端提交是post请求还是get请求：

if(req.url == '/dopost' && req.method.toLowerCase() == 'post'){}

# 二 简单的登录注册

## 1 前台

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
名称：<input type="text" id="user" /><br>  
密码：<input type="password" id="pass" /><br>  
<input type="button" value="注册" id="btn1">  
<input type="button" value="登陆" id="btn2">  
<script src="./jquery.js"></script>  
<script>  
 $(function () {  
 *//注册* $('#btn1').click(function (){  
 $.ajax({  
 url: 'http://localhost:3000/register',  
 data: {  
 user: $('#user').val(),  
 pass: $('#pass').val()  
 },  
 dataType: 'json',  
 success(data){  
 console.log(data);  
 if(data.err){  
 alert('非法.....'+ data.msg);  
 console.log(data);  
 }else{  
 alert('注册成功');  
 }  
 },  
 error(){  
 alert('错误');  
 }  
 })  
 });  
  
 *//登陆* $('#btn2').click(function (){  
 $.ajax({  
 url: 'http://localhost:3000/login',  
 data: {user: $('#user').val(), pass: $('#pass').val()},  
 dataType: 'json',  
 success(data){  
 if(data.err){  
 alert('错了，'+data.msg);  
 }else{  
 alert('登录成功');  
 }  
 },  
 error(){  
 alert('错了');  
 }  
 })  
 });  
  
 });  
</script>  
</body>  
</html>

## 2 后台

const http=require('http');  
const url=require('url');  
const querystring=require('querystring');  
const fs=require('fs');  
  
let users = {};  
  
http.createServer(function(req,res){  
  
 res.setHeader('Access-Control-Allow-Origin','\*');  
 let {pathname,query} = url.parse(req.url,true);  
 let {user,pass} = query;  
  
 *//注册* if (pathname == '/register') {  
  
 if ( !user || !pass) { *//注册的时候没填写user* res.write('{"err":1,"msg":"user or pass must be required"}');  
 } else {  
 *//存储数据* users[user] = pass;  
 console.log(users);  
 res.write('{"err": 0, "msg": "success"}');  
 }  
 res.end();  
 }  
  
 *//登录* if (pathname == '/login') {  
 if (!user || !pass) {  
 res.write('{"err":1,"msg":"user or pass must be required"}');  
 } else {  
 if (!users[user]) {  
 res.write('{"err": 1, "msg": "no such user"}');  
 } else {  
*//查询 users[user]: 用户输入的是user，传到users里查询对应的pass* if (users[user] == pass) {  
 res.write('{"err": 1, "msg": "login success"}');  
 } else {  
 res.write('{"err": 0, "msg": "user or pass is wrong"}');  
 }  
 }  
 }  
 res.end();  
 }  
  
}).listen(3000);

# 三 原生文件上传

在第一章POST请求说明中，使用了字符串变量str来接收，如果上传的是文本文件，则仍然是正确的，但是如果上传的是图片、音频等，这里就无法实现了，我们可以使用数组的形式保存：

const http = require('http');  
let server = http.createServer(function (req,res) {  
 let arr = [];  
 req.on('data',function (data) {  
 arr.push(data);  
 });  
 req.on('end',function () {  
 let buf = Buffer.concat(arr);  
 console.log(buf.toString());  
 res.end();  
 });  
});  
server.listen(8000,function () {  
 console.log('server is started...');  
});

http://student.kaikeba.com/course/63/study/1971

# 四 第三方模块formidable

一般文件上传可以使用express3.0中的connect内置中间件multipart来处理，但是这个中间件在express4.0中已经被移除，一旦express更新了对connect的依赖，它也将从express中小时。不推荐使用该中间件，推荐使用formidable。

Node需要使用模块formidable处理图片的上传。这个模块需要额外安装。另外我们将使用 silly-datetime模块为上传的文件名添加时间戳。

注意：上传文件的保存文件夹为upload

前台代码：

<div id="div">  
 <form method="post" enctype="multipart/form-data" action="http://localhost:3000/upload">  
 <input type="file" name="f">  
 <input type="submit" value="上传">  
 </form>  
</div>

后台代码：

const url = require('url');  
const http = require('http');  
const path = require('path');  
  
const formidable = require('formidable');  
  
http.createServer(function (req,res) {  
 let pathname = url.parse(req.url).pathname;  
 if (pathname == '/upload' && req.method.toLowerCase() == "post") {  
 uploading(req,res);  
 }  
}).listen(3000);  
  
*//图片上传的方法*function uploading(req,res) {  
 let form = new formidable.IncomingForm();  
 form.uploadDir = path.join(\_\_dirname, './upload');  
 form.parse(req, function (err, fields, files) {  
 if (err) throw err;  
 res.writeHead(200, {'content-type': 'text/plain;charset=utf8'});  
 res.end('上传成功');  
 });  
}

当路径名存在中文或者上传图片名包含中文时，可以正常上传，但是前台在使用时，无法展示图片信息。

这是因为url在传递的时候，浏览器会自动将请求url使用encodeURI进行转码。因此需要使用后decodeURI来解码获取中文字符，修改pathname的赋值。

let pathname = decodeURI(url.parse(req.url).pathname);