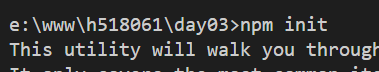
# 笔记

## 我们做一个项目：

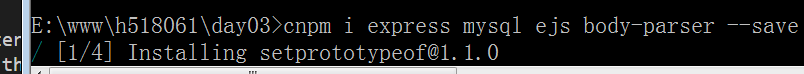
有一个文件夹，放我们的项目的代码；

我们的后续命令行操作应该在当前文件夹下；

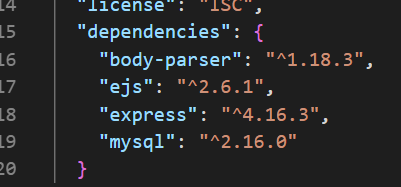


第一步，项目初始化；

npm init -y 直接生成一个项目配置文件，生成以后我们可以在编辑器里面直接修改；

第二步，安装需要的第三方模块； 

项目配置文件发生改变：



项目下面只有一个js文件作为入口文件，其他相关文件放在对应的目录下；

## 学生管理系统：

管理员登录：

1. 有一个登录页面：HTML、CSS、BootStrap；
2. 使用js进行各种检查以及ajax处理：JS、jQuery；
3. 后台数据处理：nodejs、Express、MySQL；

MVC：架构思想

## 修改信息：

<a href="/updstu?sid=<%=s.sid%>" ~~sid="<%=s.sid%>"~~>修改</a>

1. 我们应该知道修改的是哪一条记录：传主键过去；
2. 有一个表单（页面）让用户去修改信息：显示出来原始信息；
3. 提交，把数据更新到数据库：我要知道更新的是哪一条记录；

## AJAX:

异步；

实现局部刷新：

不存在的账号，可以及时提示，并且当前页面不会刷新，提示用户体验；

传递的数据很少，减轻服务器压力，节省带宽；

## Nodejs：非阻塞

改成阻塞的：

第三方模块 ： async 可以把异步变成同步

//实现同步 参数一 任务：object 参数二 任务执行完成后的操作：fn

async.series({

work2000: function (callback) {

fs.readFile('./a.txt', (err, str) => {

//结构性要求：任务函数必须有一回调函数 如果没有callback 任务不会接着向下执行

callback(null, str.toString());

});

},

work2: function (cb) {

fs.readFile('./b.txt', (err, str) => {

cb(null, '刘桐');

});

},

work3: function (cb) {

fs.readFile('./c.txt', (err, str) => {

cb(null,[1,2,3,4,{a:1,b:2}]);

});

}

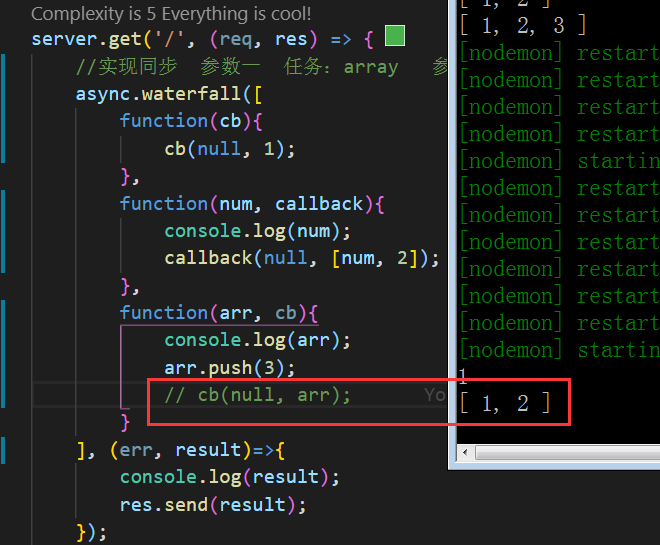
}, (err, result) => {

console.log(result);

res.send(result);

});

瀑布流;



## 分页：

1. 要确定每页显示多少条：pagenum=50;
2. 要知道当前是第几页：page=1

SELECT \* FROM table LIMIT start, num;

第一页： limit 0, pagenum

第二页：limit 50, pagenum

第n页：limit (n-1)\*pagenum, pagenum

上一页：当前页数-1 不能小于1

下一页：当前页数\*1+1 不能大于总页数

最多显示5页：Showpage = 5

6 7 **8** 9 10

85 86 **87** 88 89

For(I = start; I <= end; i++){

}

数据关系：

End - start = Showpage – 1

Start = page –（Showpage – 1）/2; 开始页数

End = page + （Showpage – 1）/2; 结束页数