

安装nodejs 全局安装nodemon

安装mysql 添加环境变量，登陆系统

Mysql –uroot –p

创建数据库 create safety

创建表，可以从文件中一次性创建 mysql safety –uroot < safety01.sql 一次性建立了表

项目结构

安装nodemon

这样有代码变更就会自动重启服务而不需要npm start

框架是koa2, 模板是pug,也就是jade的前身；

看入口文件 server.js 看代码注释

Views里有模板文件

Router里是路由配置，紧密结合的是controller,

Model里是数据，有链接数据库的操作

Static一些静态资源

用jspm前端包管理工具 jspm install

安装前端依赖 npm install 管理后端node依赖

先jspm install 安装前段依赖

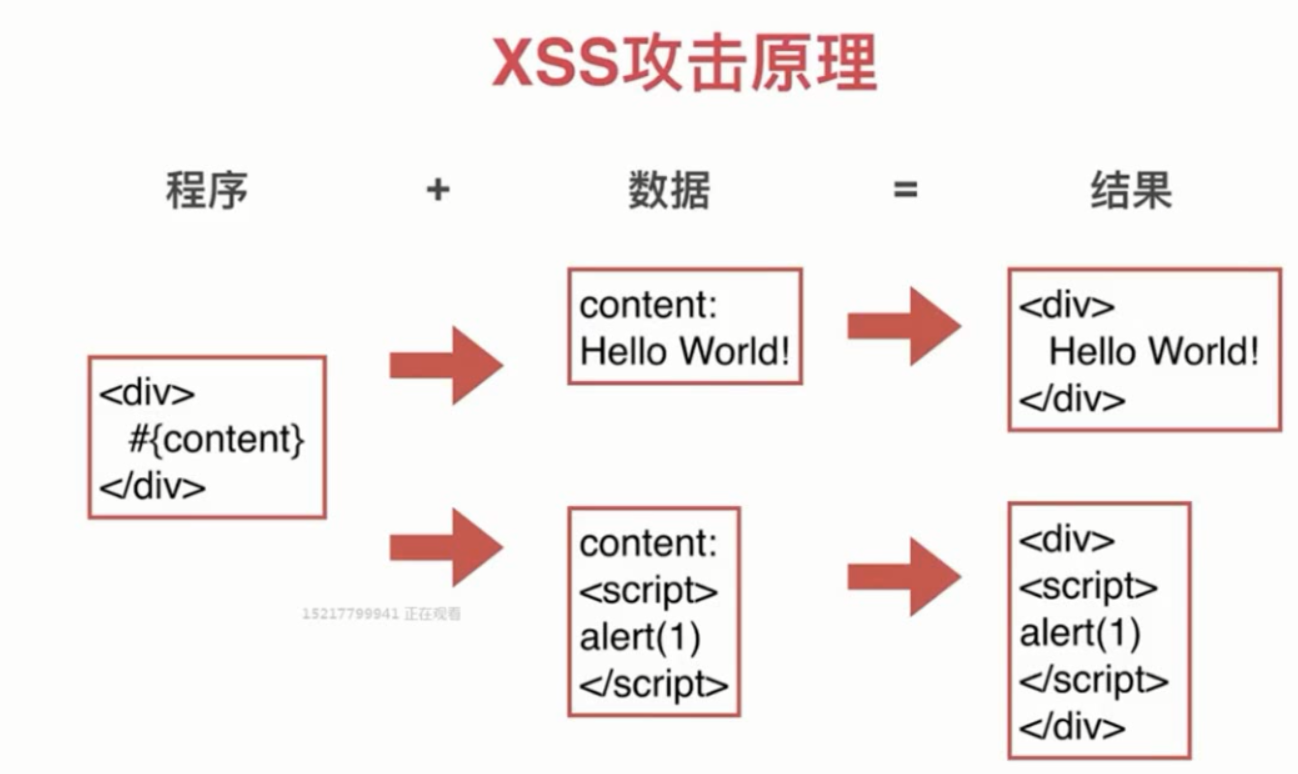
在 npm install 安装后端以来

Npm run dev

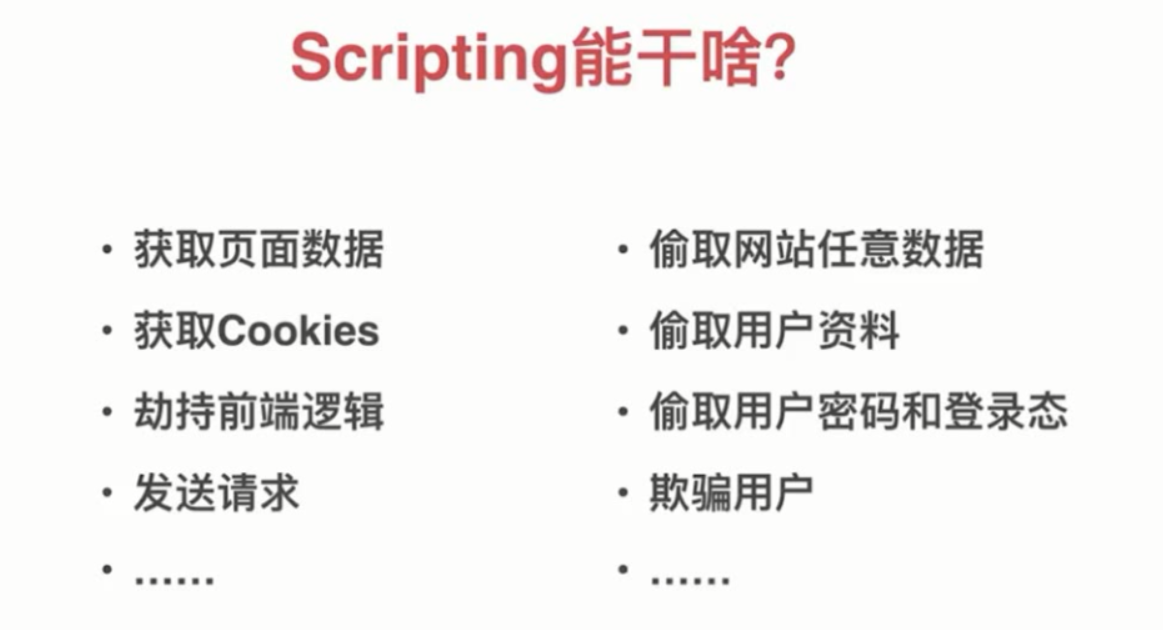
"dev": "NODE\_ENV=development nodemon server.js -e\"js pug\"",



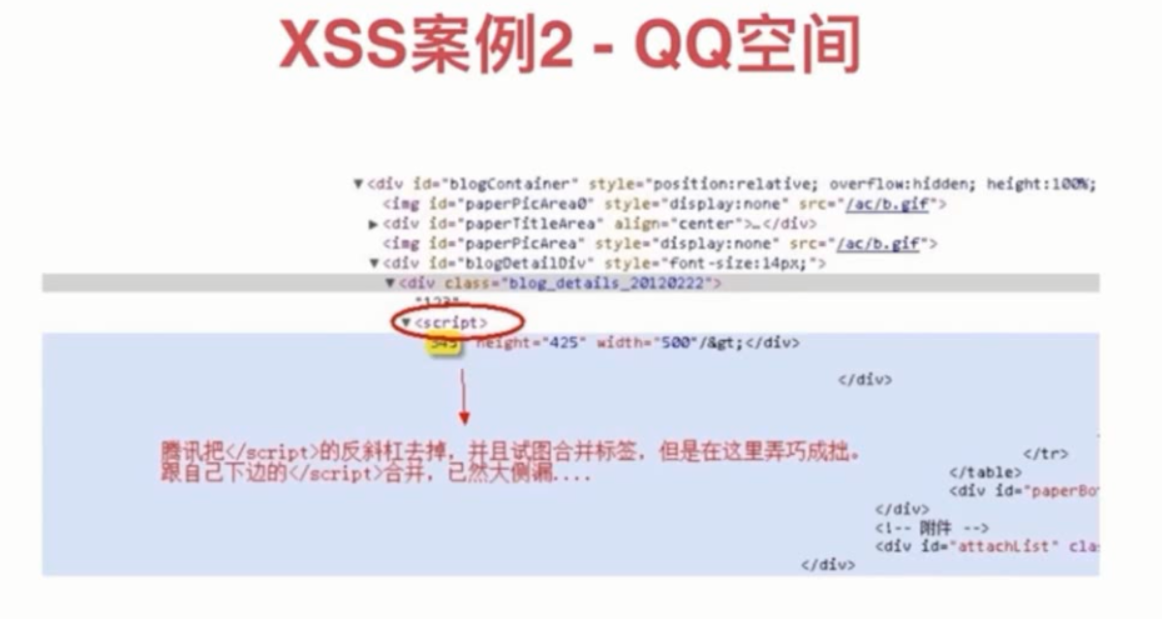
为啥叫xss ,css – c-x

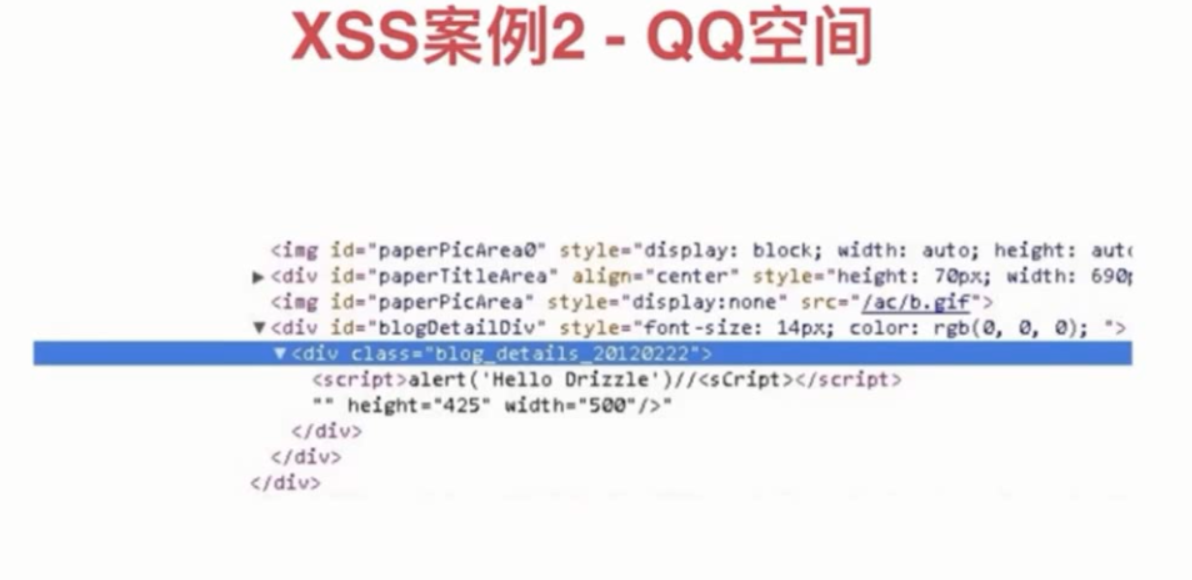


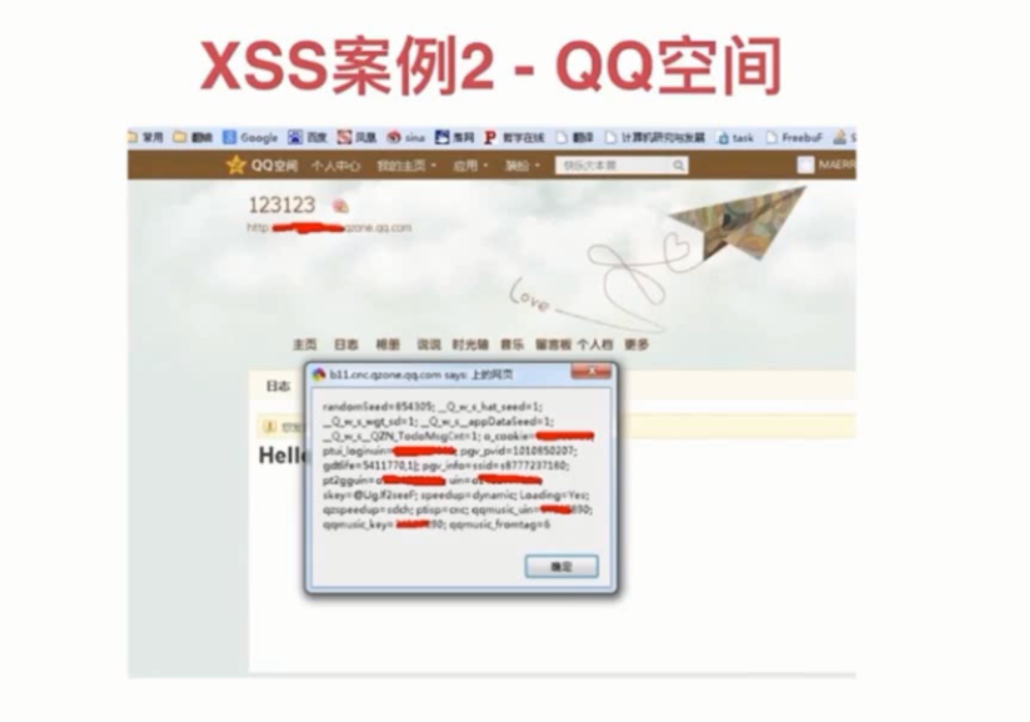


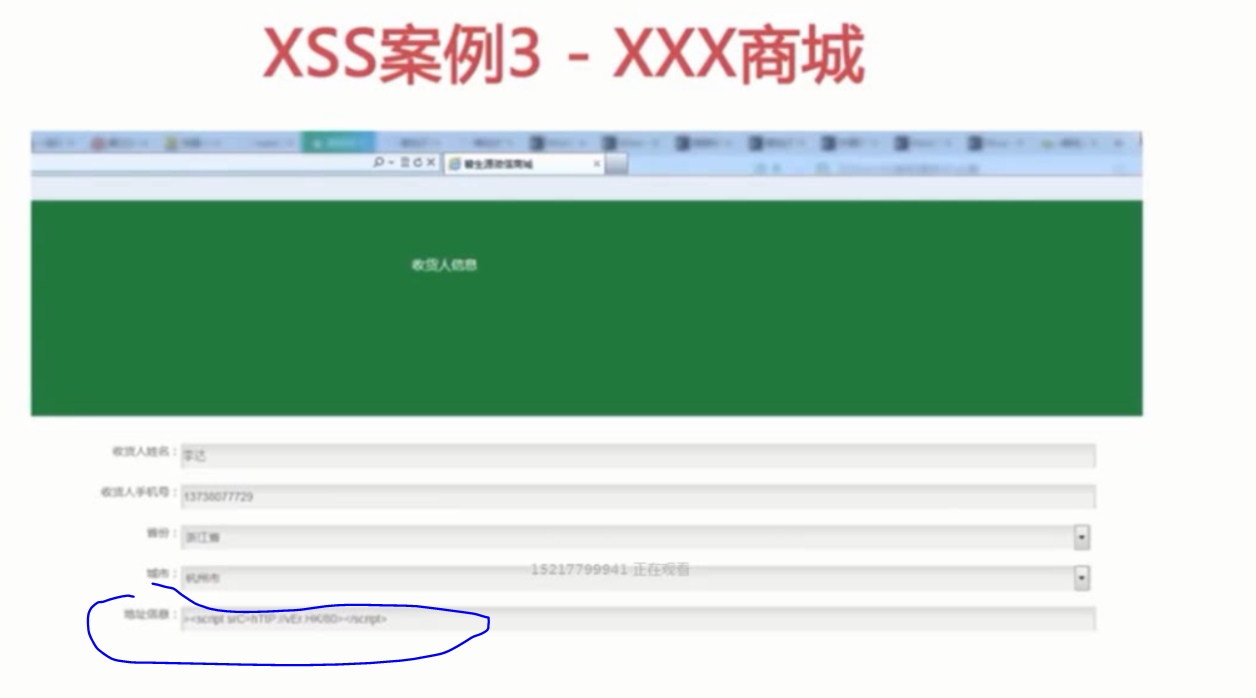




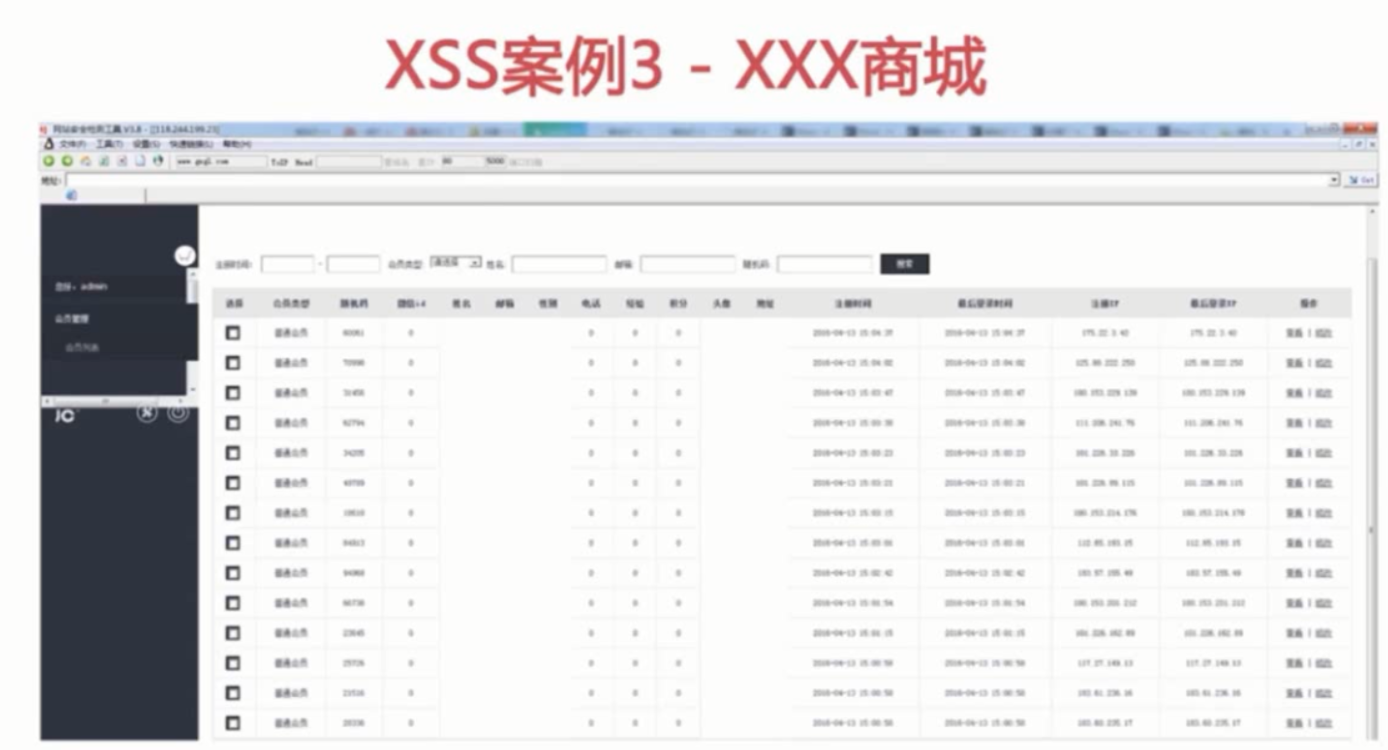












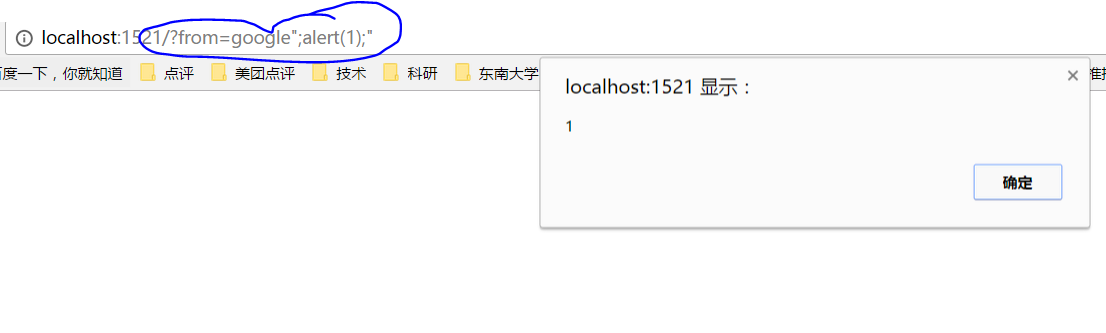


Js代码的攻击，主要是由于一些变量被特殊字符(如””)包裹起来的，那么用户有可能会用“”，来提前结束这些变量，或者语句。

解决方案，只需将这些特殊符号做一些转义就好了；



**fromForJs**: ctx.query.from, //直接不过滤



可以将“” 和 // 过滤掉

**var** *escapeForJs* = **function**(str) {  
 **if**(!str) **return** '';  
 str = str.replace(/\\/g, '\\\\');  
 str = str.replace(/"/g, '\\"');  
 **return** str;  
};

不过很难取尽，所以最安全的还是用对输入的进行字符串序列化处理

JSON.stringify(ctx.query.from)

富文本 html格式的字符

黑名单过滤 <script 标签 onerror 属性